EXHIBIT 13

IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
WACO DIVISION

WSOU INVESTMENTS, LLC D/B/A	**§**	
BRAZOS LICENSING AND	**§**	**Case No. 6:20-cv-00957-ADA**
DEVELOPMENT,	**§**	
Plaintiff,	**§**	**JURY TRIAL DEMANDED**
	§	
v.	**§**	
	§	
ONEPLUS TECHNOLOGY	**§**	
(SHENZHEN) CO., LTD.,	**§**	
Defendant.	**§**	

PLAINTIFF’S AMENDED FINAL DISCLOSURES OF PRELIMINARY INFRINGEMENT CONTENTIONS

Pursuant to the Court’s Order Governing Proceeding – Patent Case (“Order Governing Proceeding”), Plaintiff WSOU Investments, LLC d/b/a Brazos Licensing and Development (“WSOU”) hereby provides its Amended Final Infringements Contentions to defendant OnePlus Technology (Shenzhen) Co., Ltd. (“OnePlus” or “Defendant”) for U.S. Patent No. 8,712,708 (the “’708 Patent”).

WSOU makes this disclosure based on the information presently available to it. Discovery in this case has not started, and WSOU reserves its right to amend or supplement these disclosures as permitted by the Federal Rules of Civil Procedure, by the local rules of the Western District of Texas, and by order of the Court, including the Court’s Order Governing Proceedings.

For each Asserted Claim, Plaintiff identifies the following Accused Instrumentalities of which it is currently aware. The identification of Accused Instrumentalities is based on Plaintiff’s research and analysis to date, without the benefit of discovery from the Defendant. Plaintiff reserves the right to add, delete, substitute or otherwise amend this list of Accused

Instrumentalities based on discovery or other circumstances, in a manner consistent with the Federal Rules of Civil Procedures, local rules, and standing orders.

The Accused Instrumentalities include, without limitation, the following:

- OnePlus Mobile Phones (such as OnePlus 8, OnePlus 7 Pro, OnePlus 8 Pro, OnePlus 8T, OnePlus 9, OnePlus 9 Pro).
- All past, current and future OnePlus products and services that operate in the same or substantially similar manner as the specifically identified products and services above and described in Exhibit 1.
- All past, current and future OnePlus products and services that have the same or substantially similar features as the specifically identified products and services above and described in Exhibit 1.

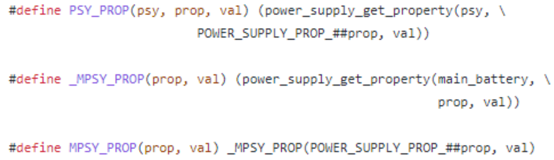
Plaintiff's infringement contentions apply to the Accused Instrumentalities as well as all other past, current and future hardware and software products and services developed, made, used, offered for sale, sold, imported, and provided by OnePlus that contain or makes use of the Patented Technology.[1]

Based upon currently available information, WSOU asserts that OnePlus has infringed and/or continues to infringe the patent and claims identified in the attached claim charts (the "Asserted Claims" of the "Patent-in-Suit"). Infringement claim charts evidencing the correspondence between (i) the elements of the Asserted Claims, and (ii) the corresponding items of the accused products are attached hereto. Further, Exhibit 1, which is attached hereto and incorporated by reference, is an exemplary infringement claim chart identifying specifically where

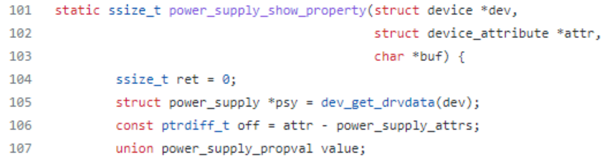
[1] "Patented Technology" means all technologies described in the claims of the Patent-in-Suit.

each limitation of each Asserted Claim is found within each Accused Instrumentality or practiced by each Accused Instrumentality.

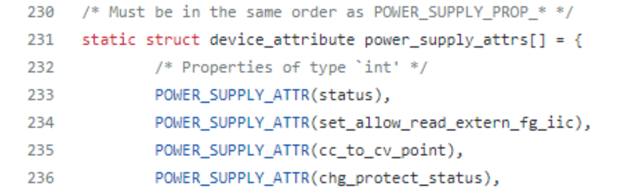
Plaintiff asserts that Defendant has directly infringed and continues to directly infringe the Asserted Claims literally through the Accused Instrumentalities by making, using, offering for sale, and/or selling, or importing into the United States the Accused Instrumentalities. To the extent that Defendant alleges that one or more limitations of the Asserted Claims are not literally found in the Accused Instrumentalities, Plaintiff alleges that such limitations are found in or practiced by the Accused Instrumentalities under the doctrine of equivalents. Any differences alleged to exist between any of the Asserted Claims and any of the Accused Instrumentalities are insubstantial and that each Accused Instrumentality also meets each limitation under the doctrine of equivalents as the identified features of the Accused Instrumentality performs substantially the same function in substantially the same way to achieve substantially the same result as the corresponding claim limitation. WSOU reserves the right to assert infringement solely under the doctrine of equivalents with respect to any particular claim element(s), if warranted by discovery, further analysis, and/or claim constructions in this case.

<table>
<tr><th><u>SM8150</u>
<u>(OnePlus 7 Pro)</u></th></tr>
<tr><td>

<pre>
#define PSY_PROP(psy, prop, val) (power_supply_get_property(psy, \
                        POWER_SUPPLY_PROP_##prop, val))

#define _MPSY_PROP(prop, val) (power_supply_get_property(main_battery, \
                                                  prop, val))

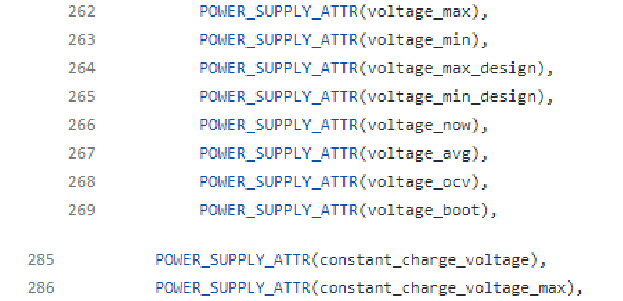
#define MPSY_PROP(prop, val) _MPSY_PROP(POWER_SUPPLY_PROP_##prop, val)
</pre>
<u>Source</u>
</td></tr>
<tr><td>

<pre>
static ssize_t power_supply_show_property(struct device *dev,
                                          struct device_attribute *attr,
                                          char *buf) {
        ssize_t ret = 0;
        struct power_supply *psy = dev_get_drvdata(dev);
        const ptrdiff_t off = attr - power_supply_attrs;
        union power_supply_propval value;
</pre>
</td></tr>
</table>

Source



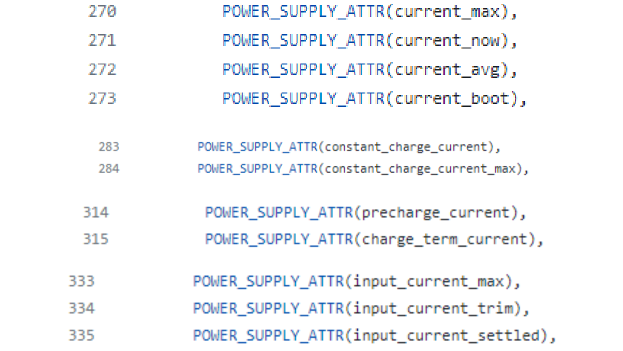
```
/* Must be in the same order as POWER_SUPPLY_PROP_* */
static struct device_attribute power_supply_attrs[] = {
        /* Properties of type `int' */
        POWER_SUPPLY_ATTR(status),
        POWER_SUPPLY_ATTR(set_allow_read_extern_fg_iic),
        POWER_SUPPLY_ATTR(cc_to_cv_point),
        POWER_SUPPLY_ATTR(chg_protect_status),
```

Source



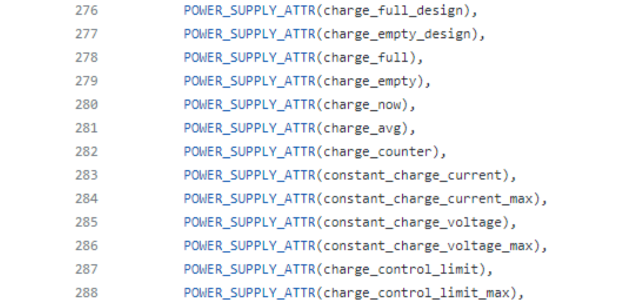
```
        POWER_SUPPLY_ATTR(voltage_max),
        POWER_SUPPLY_ATTR(voltage_min),
        POWER_SUPPLY_ATTR(voltage_max_design),
        POWER_SUPPLY_ATTR(voltage_min_design),
        POWER_SUPPLY_ATTR(voltage_now),
        POWER_SUPPLY_ATTR(voltage_avg),
        POWER_SUPPLY_ATTR(voltage_ocv),
        POWER_SUPPLY_ATTR(voltage_boot),

        POWER_SUPPLY_ATTR(constant_charge_voltage),
        POWER_SUPPLY_ATTR(constant_charge_voltage_max),
```

Source



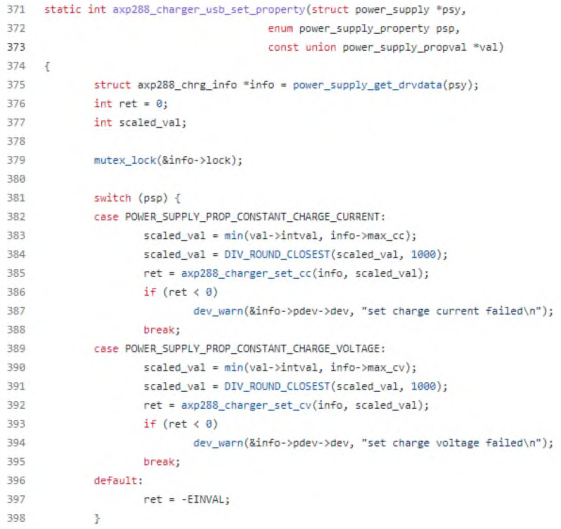
```
        POWER_SUPPLY_ATTR(current_max),
        POWER_SUPPLY_ATTR(current_now),
        POWER_SUPPLY_ATTR(current_avg),
        POWER_SUPPLY_ATTR(current_boot),

        POWER_SUPPLY_ATTR(constant_charge_current),
        POWER_SUPPLY_ATTR(constant_charge_current_max),

        POWER_SUPPLY_ATTR(precharge_current),
        POWER_SUPPLY_ATTR(charge_term_current),

        POWER_SUPPLY_ATTR(input_current_max),
        POWER_SUPPLY_ATTR(input_current_trim),
        POWER_SUPPLY_ATTR(input_current_settled),
```

Source



```
        POWER_SUPPLY_ATTR(charge_full_design),
        POWER_SUPPLY_ATTR(charge_empty_design),
        POWER_SUPPLY_ATTR(charge_full),
        POWER_SUPPLY_ATTR(charge_empty),
        POWER_SUPPLY_ATTR(charge_now),
        POWER_SUPPLY_ATTR(charge_avg),
        POWER_SUPPLY_ATTR(charge_counter),
        POWER_SUPPLY_ATTR(constant_charge_current),
        POWER_SUPPLY_ATTR(constant_charge_current_max),
        POWER_SUPPLY_ATTR(constant_charge_voltage),
        POWER_SUPPLY_ATTR(constant_charge_voltage_max),
        POWER_SUPPLY_ATTR(charge_control_limit),
        POWER_SUPPLY_ATTR(charge_control_limit_max),
```

Source

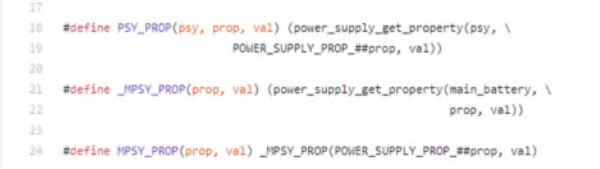


```
static int axp288_charger_usb_set_property(struct power_supply *psy,
                                    enum power_supply_property psp,
                                    const union power_supply_propval *val)
{
        struct axp288_chrg_info *info = power_supply_get_drvdata(psy);
        int ret = 0;
        int scaled_val;

        mutex_lock(&info->lock);

        switch (psp) {
        case POWER_SUPPLY_PROP_CONSTANT_CHARGE_CURRENT:
                scaled_val = min(val->intval, info->max_cc);
                scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
                ret = axp288_charger_set_cc(info, scaled_val);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set charge current failed\n");
                break;
        case POWER_SUPPLY_PROP_CONSTANT_CHARGE_VOLTAGE:
                scaled_val = min(val->intval, info->max_cv);
                scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
                ret = axp288_charger_set_cv(info, scaled_val);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set charge voltage failed\n");
                break;
        default:
                ret = -EINVAL;
        }
```

Source

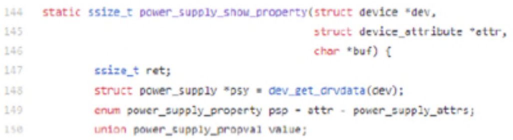
SM8250

(OnePlus 8, 8 Pro, 8T)



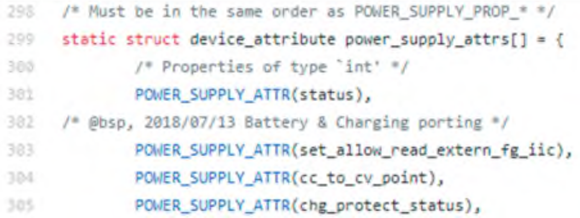
```

#define PSY_PROP(psy, prop, val) (power_supply_get_property(psy, \
                        POWER_SUPPLY_PROP_##prop, val))

#define _MPSY_PROP(prop, val) (power_supply_get_property(main_battery, \
                                                prop, val))

#define MPSY_PROP(prop, val) _MPSY_PROP(POWER_SUPPLY_PROP_##prop, val)
```

Source: Exhibit G



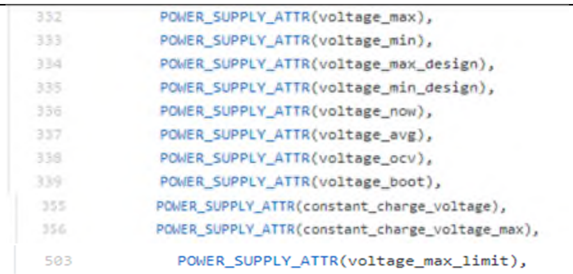
```
static ssize_t power_supply_show_property(struct device *dev,
                                          struct device_attribute *attr,
                                          char *buf) {
        ssize_t ret;
        struct power_supply *psy = dev_get_drvdata(dev);
        enum power_supply_property psp = attr - power_supply_attrs;
        union power_supply_propval value;
```

Source: Exhibit E



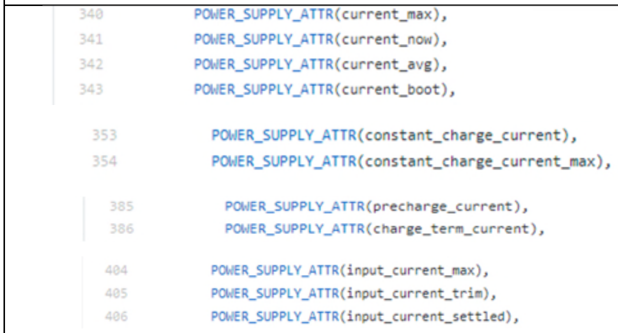
```
/* Must be in the same order as POWER_SUPPLY_PROP_* */
static struct device_attribute power_supply_attrs[] = {
        /* Properties of type `int' */
        POWER_SUPPLY_ATTR(status),
/* @bsp, 2018/07/13 Battery & Charging porting */
        POWER_SUPPLY_ATTR(set_allow_read_extern_fg_iic),
        POWER_SUPPLY_ATTR(cc_to_cv_point),
        POWER_SUPPLY_ATTR(chg_protect_status),
```

Source: Exhibit E



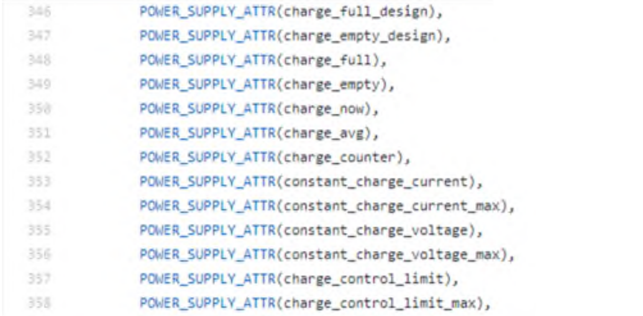
```
	POWER_SUPPLY_ATTR(voltage_max),
	POWER_SUPPLY_ATTR(voltage_min),
	POWER_SUPPLY_ATTR(voltage_max_design),
	POWER_SUPPLY_ATTR(voltage_min_design),
	POWER_SUPPLY_ATTR(voltage_now),
	POWER_SUPPLY_ATTR(voltage_avg),
	POWER_SUPPLY_ATTR(voltage_ocv),
	POWER_SUPPLY_ATTR(voltage_boot),
	POWER_SUPPLY_ATTR(constant_charge_voltage),
	POWER_SUPPLY_ATTR(constant_charge_voltage_max),

	POWER_SUPPLY_ATTR(voltage_max_limit),
```

Source: Exhibit E



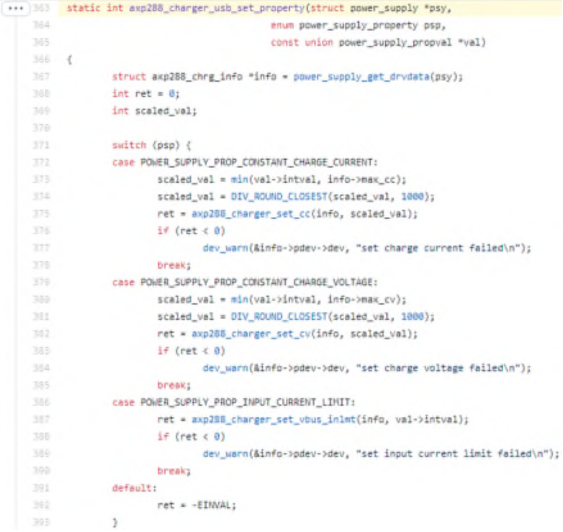
```
	POWER_SUPPLY_ATTR(current_max),
	POWER_SUPPLY_ATTR(current_now),
	POWER_SUPPLY_ATTR(current_avg),
	POWER_SUPPLY_ATTR(current_boot),

	POWER_SUPPLY_ATTR(constant_charge_current),
	POWER_SUPPLY_ATTR(constant_charge_current_max),

	POWER_SUPPLY_ATTR(precharge_current),
	POWER_SUPPLY_ATTR(charge_term_current),

	POWER_SUPPLY_ATTR(input_current_max),
	POWER_SUPPLY_ATTR(input_current_trim),
	POWER_SUPPLY_ATTR(input_current_settled),
```

Source: Exhibit E



```
	POWER_SUPPLY_ATTR(charge_full_design),
	POWER_SUPPLY_ATTR(charge_empty_design),
	POWER_SUPPLY_ATTR(charge_full),
	POWER_SUPPLY_ATTR(charge_empty),
	POWER_SUPPLY_ATTR(charge_now),
	POWER_SUPPLY_ATTR(charge_avg),
	POWER_SUPPLY_ATTR(charge_counter),
	POWER_SUPPLY_ATTR(constant_charge_current),
	POWER_SUPPLY_ATTR(constant_charge_current_max),
	POWER_SUPPLY_ATTR(constant_charge_voltage),
	POWER_SUPPLY_ATTR(constant_charge_voltage_max),
	POWER_SUPPLY_ATTR(charge_control_limit),
	POWER_SUPPLY_ATTR(charge_control_limit_max),
```

Source: Exhibit E

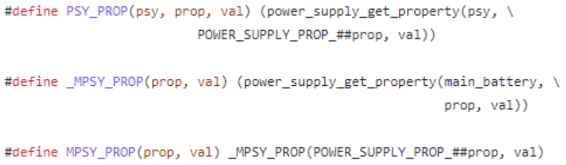


```
static int axp288_charger_usb_set_property(struct power_supply *psy,
				    enum power_supply_property psp,
				    const union power_supply_propval *val)
{
	struct axp288_chrg_info *info = power_supply_get_drvdata(psy);
	int ret = 0;
	int scaled_val;

	switch (psp) {
	case POWER_SUPPLY_PROP_CONSTANT_CHARGE_CURRENT:
		scaled_val = min(val->intval, info->max_cc);
		scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
		ret = axp288_charger_set_cc(info, scaled_val);
		if (ret < 0)
			dev_warn(&info->pdev->dev, "set charge current failed\n");
		break;
	case POWER_SUPPLY_PROP_CONSTANT_CHARGE_VOLTAGE:
		scaled_val = min(val->intval, info->max_cv);
		scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
		ret = axp288_charger_set_cv(info, scaled_val);
		if (ret < 0)
			dev_warn(&info->pdev->dev, "set charge voltage failed\n");
		break;
	case POWER_SUPPLY_PROP_INPUT_CURRENT_LIMIT:
		ret = axp288_charger_set_vbus_inlmt(info, val->intval);
		if (ret < 0)
			dev_warn(&info->pdev->dev, "set input current limit failed\n");
		break;
	default:
		ret = -EINVAL;
	}
```

Source: Exhibit H

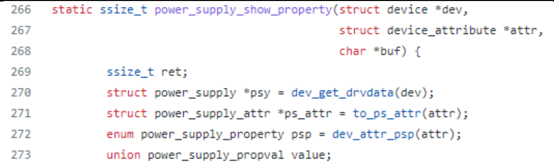
SM8350

(OnePlus 9, 9 Pro)



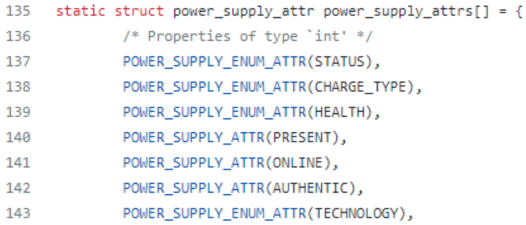
```
#define PSY_PROP(psy, prop, val) (power_supply_get_property(psy, \
                        POWER_SUPPLY_PROP_##prop, val))

#define _MPSY_PROP(prop, val) (power_supply_get_property(main_battery, \
                                                prop, val))

#define MPSY_PROP(prop, val) _MPSY_PROP(POWER_SUPPLY_PROP_##prop, val)
```

Source



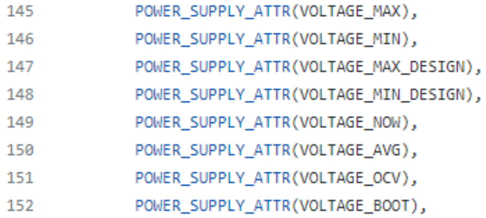
```
static ssize_t power_supply_show_property(struct device *dev,
                                          struct device_attribute *attr,
                                          char *buf) {
        ssize_t ret;
        struct power_supply *psy = dev_get_drvdata(dev);
        struct power_supply_attr *ps_attr = to_ps_attr(attr);
        enum power_supply_property psp = dev_attr_psp(attr);
        union power_supply_propval value;
```

Source

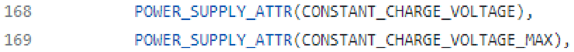


```
static struct power_supply_attr power_supply_attrs[] = {
        /* Properties of type `int' */
        POWER_SUPPLY_ENUM_ATTR(STATUS),
        POWER_SUPPLY_ENUM_ATTR(CHARGE_TYPE),
        POWER_SUPPLY_ENUM_ATTR(HEALTH),
        POWER_SUPPLY_ATTR(PRESENT),
        POWER_SUPPLY_ATTR(ONLINE),
        POWER_SUPPLY_ATTR(AUTHENTIC),
        POWER_SUPPLY_ENUM_ATTR(TECHNOLOGY),
```

Source

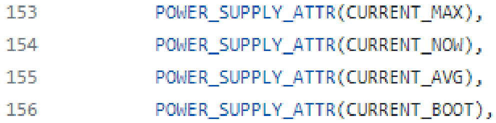


```
        POWER_SUPPLY_ATTR(VOLTAGE_MAX),
        POWER_SUPPLY_ATTR(VOLTAGE_MIN),
        POWER_SUPPLY_ATTR(VOLTAGE_MAX_DESIGN),
        POWER_SUPPLY_ATTR(VOLTAGE_MIN_DESIGN),
        POWER_SUPPLY_ATTR(VOLTAGE_NOW),
        POWER_SUPPLY_ATTR(VOLTAGE_AVG),
        POWER_SUPPLY_ATTR(VOLTAGE_OCV),
        POWER_SUPPLY_ATTR(VOLTAGE_BOOT),
```

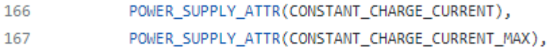


```
        POWER_SUPPLY_ATTR(CONSTANT_CHARGE_VOLTAGE),
        POWER_SUPPLY_ATTR(CONSTANT_CHARGE_VOLTAGE_MAX),
```

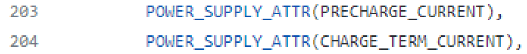
Source



```
        POWER_SUPPLY_ATTR(CURRENT_MAX),
        POWER_SUPPLY_ATTR(CURRENT_NOW),
        POWER_SUPPLY_ATTR(CURRENT_AVG),
        POWER_SUPPLY_ATTR(CURRENT_BOOT),
```

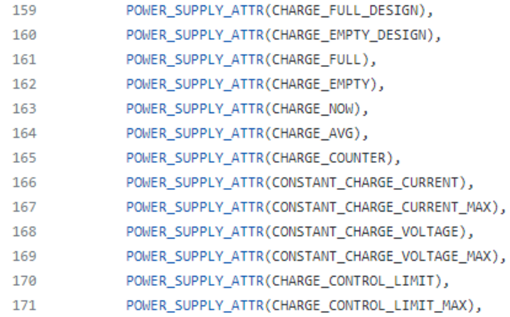


```
        POWER_SUPPLY_ATTR(CONSTANT_CHARGE_CURRENT),
        POWER_SUPPLY_ATTR(CONSTANT_CHARGE_CURRENT_MAX),
```



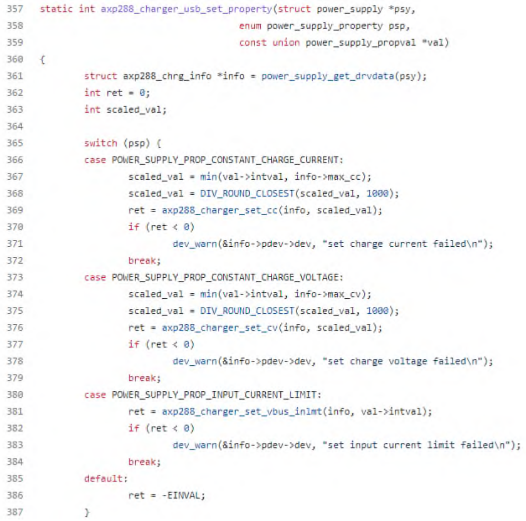
```
	POWER_SUPPLY_ATTR(PRECHARGE_CURRENT),
	POWER_SUPPLY_ATTR(CHARGE_TERM_CURRENT),
```

Source



```
	POWER_SUPPLY_ATTR(CHARGE_FULL_DESIGN),
	POWER_SUPPLY_ATTR(CHARGE_EMPTY_DESIGN),
	POWER_SUPPLY_ATTR(CHARGE_FULL),
	POWER_SUPPLY_ATTR(CHARGE_EMPTY),
	POWER_SUPPLY_ATTR(CHARGE_NOW),
	POWER_SUPPLY_ATTR(CHARGE_AVG),
	POWER_SUPPLY_ATTR(CHARGE_COUNTER),
	POWER_SUPPLY_ATTR(CONSTANT_CHARGE_CURRENT),
	POWER_SUPPLY_ATTR(CONSTANT_CHARGE_CURRENT_MAX),
	POWER_SUPPLY_ATTR(CONSTANT_CHARGE_VOLTAGE),
	POWER_SUPPLY_ATTR(CONSTANT_CHARGE_VOLTAGE_MAX),
	POWER_SUPPLY_ATTR(CHARGE_CONTROL_LIMIT),
	POWER_SUPPLY_ATTR(CHARGE_CONTROL_LIMIT_MAX),
```

Source



```
static int axp288_charger_usb_set_property(struct power_supply *psy,
					enum power_supply_property psp,
					const union power_supply_propval *val)
{
	struct axp288_chrg_info *info = power_supply_get_drvdata(psy);
	int ret = 0;
	int scaled_val;

	switch (psp) {
	case POWER_SUPPLY_PROP_CONSTANT_CHARGE_CURRENT:
		scaled_val = min(val->intval, info->max_cc);
		scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
		ret = axp288_charger_set_cc(info, scaled_val);
		if (ret < 0)
			dev_warn(&info->pdev->dev, "set charge current failed\n");
		break;
	case POWER_SUPPLY_PROP_CONSTANT_CHARGE_VOLTAGE:
		scaled_val = min(val->intval, info->max_cv);
		scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
		ret = axp288_charger_set_cv(info, scaled_val);
		if (ret < 0)
			dev_warn(&info->pdev->dev, "set charge voltage failed\n");
		break;
	case POWER_SUPPLY_PROP_INPUT_CURRENT_LIMIT:
		ret = axp288_charger_set_vbus_inlmt(info, val->intval);
		if (ret < 0)
			dev_warn(&info->pdev->dev, "set input current limit failed\n");
		break;
	default:
		ret = -EINVAL;
	}
```

Source

Plaintiff further asserts that Defendant has indirectly infringed and continues to indirectly infringe by actively inducing infringement of one or more of the claims of the Asserted Patent through the Accused Instrumentalities. Plaintiff also asserts that these third-parties directly infringe at least one or more of the claims of the Asserted Patent through the manufacture, use, sale, offer to sell, or importation of the Accused Instrumentalities.

For example, Defendant has actively induced infringement by encouraging the use of the Accused Instrumentalities in ways that infringe each Asserted Claim, including, but not limited through providing instructions to its customers and partners to encourage and instruct the user or partner to utilize the accused product in an infringing manner. Defendant knew or should have known that such encouragement would induce infringement. Defendant has taken active steps with the specific intent to encourage and cause others to use each Accused Instrumentality in ways that infringe each Asserted Claim. Such active steps by Defendant with specific intent to induce infringement have included, among other things, advertising, promoting, marketing, making available for use, offering to sell, and/or selling the Accused Instrumentalities to others; encouraging and influencing others to import, offer to sell, and/or sell the Accused Instrumentalities; directing and instructing others to use the Accused Instrumentalities in infringing ways; and by providing the Accused Instrumentalities to others. OnePlus has performed the aforementioned active steps with the knowledge of the Asserted Patent at least as of the date when the complaint in this case was filed. OnePlus has known or should have known that the acts it has induced constitute infringement because, for instance, it has been aware that end users and resellers will purchase the Accused Instrumentalities will use them, resulting in direct infringement.

Further, for instance, the Accused Instrumentalities are known by Defendant to be especially made or especially adapted for use to infringe the Asserted Patent, and are not staple articles or commodity of commerce suitable for substantial non-infringing uses. Defendant contributes to the infringement of the Asserted Patent by making available for use, offering for sale, selling, and/or importing the Accused Instrumentalities to third parties, who use the Accused Instrumentalities and/or practice one or more claims of the Asserted Patent. Moreover, Defendant has had notice of the Asserted Patent at least as of the filing of the Complaint in this case.

These Infringement Contentions, including Exhibit 1, are based upon publicly-available information, and Plaintiff's research and analysis to date. The Accused Instrumentalities involve confidential, proprietary designs that are not publicly available, and Defendant has not yet provided discovery. Discovery is ongoing, and Plaintiff anticipates that the subject matter of these infringement contentions will be the subject of expert discovery. Discovery will provide evidence of Defendant's infringement, may lead to the discovery of additional instances of infringement, and may also enable identification of additional claims that are infringed by Defendant. Plaintiff reserves the right to add, delete, substitute, or otherwise further amend these Infringement Contentions based on discovery or other circumstances, in a manner consistent with the Federal Rules of Civil Procedures, local rules, and standing orders. Plaintiff explicitly reserves the right to further modify and/or supplement these contentions with additional or different theories and/or additional or different evidence. Further, WSOU reserves the right to supplement or revise its infringement contentions and/or chart, including identification of additional asserted claims, based on, for example, new versions or variations of one or more of the Accused Instrumentalities that are later discovered.

PRIORITY DATE

Each of the Asserted Claims of the '708 Patent is entitled to a priority date of no later than Oct. 25, 2010. The subject matter described by the Asserted Claims, however, may have been conceived and reduced to practice prior to this priority date. WSOU also reserves the right to identify any portions of the file history as containing evidence of conception and reduction to practice. Plaintiff's research and analysis is ongoing and Plaintiff reserves the right to assert that the claims are entitled to a priority date that is earlier than the above date.

Dated: October 26, 2021

RESPECTFULLY SUBMITTED,

By: */s/ Jonathan K. Waldrop*

Jonathan K. Waldrop (CA Bar No. 297903)
(Admitted in this District)
jwaldrop@kasowitz.com
Darcy L. Jones (CA Bar No. 309474)
(Admitted in this District)
djones@kasowitz.com
Marcus A. Barber (CA Bar No. 307361)
(Admitted in this District)
mbarber@kasowitz.com
John W. Downing (CA Bar No. 252850)
(Admitted in this District)
jdowning@kasowitz.com
Heather S. Kim (CA Bar No. 277686)
(Admitted in this District)
hkim@kasowitz.com
Jack Shaw (CA Bar No. 309382)
(Admitted in this District)
jshaw@kasowitz.com
KASOWITZ BENSON TORRES LLP
333 Twin Dolphin Drive, Suite 200
Redwood Shores, California 94065
Telephone: (650) 453-5170
Facsimile: (650) 453-5171

Joshua A. Whitehill (NY Bar No. 4766473)
jwhitehill@kasowitz.com
(*Pro hac vice* admission)
Bradley P. Lerman (NY Bar No. 4906079)
blerman@kasowitz.com
(*Pro hac vice* admission)
Howard L. Bressler (NY Bar No. 248379)
hbressler@kasowitz.com
(*Pro hac vice* admission)
KASOWITZ BENSON TORRES LLP
1633 Broadway
New York, NY 10019
Telephone: (212) 506-1700
Facsimile: (212) 506-1800

Mark D. Siegmund (TX Bar No. 24117055)
mark@swclaw.com
STECKLER WAYNE COCHRAN CHERRY PLLC
8416 Old McGregor Road
Waco, TX 76712
Telephone: (254) 651-3690
Facsimile: (972) 387-4041

Attorneys for Plaintiff
WSOU INVESTMENTS, LLC d/b/a
BRAZOS LICENSING AND
DEVELOPMENT

CERTIFICATE OF SERVICE

A true and correct copy of the foregoing instrument was served or delivered electronically to all counsel of record, on this 26th day of October, 2021.

/s/ Jonathan K. Waldrop
Jonathan K. Waldrop

Exhibit 1 to
WSOU Investments, LLC's
Amended Preliminary Infringement Contentions

Infringement Claim Chart of U.S. Patent No. 8,712,708 (the "Asserted Patent")

The Accused Instrumentalities include, without limitation, OnePlus Technology (Shenzhen) Co., Ltd. ("OnePlus" or "Defendant"), OnePlus Mobile Phones such as OnePlus 8, OnePlus 7 Pro, OnePlus 8 Pro, OnePlus 8T, OnePlus 9, OnePlus 9 Pro; all past, current and future OnePlus products and services that operate in the same or substantially similar manner as the specifically identified products and services; and all past, current and future OnePlus products and services that have the same or substantially similar features as the specifically identified products and services.

WSOU Investments, LLC ("WSOU" or "Plaintiff") contends that OnePlus, including OnePlus's employees, directly infringes each of the Asserted Claims, either literally or under the doctrine of equivalents. WSOU also contends that OnePlus has indirectly infringed and continues to indirectly infringe by contributing to and actively inducing infringement of one or more of the Asserted Claims.

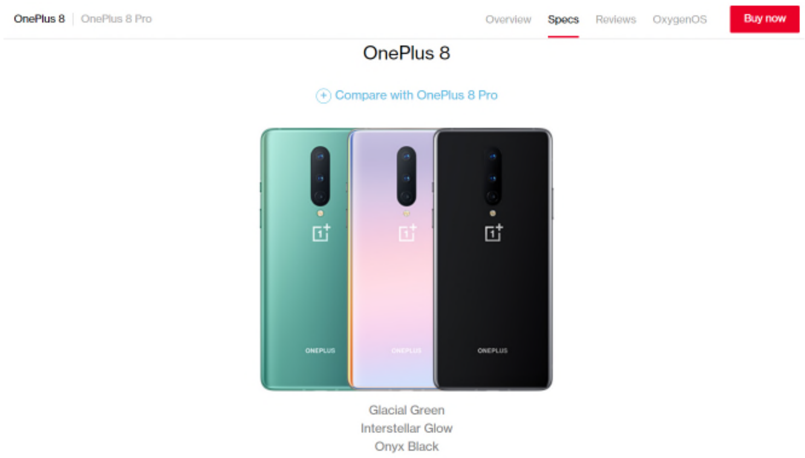
WSOU does not intend this exemplary claim chart to be limiting, and WSOU reserves its rights to pursue other accused instrumentalities, patent claims, evidence, and infringement arguments in this case.

Exhibit(s)	**Description**	**Link**
Exhibit A	OnePlus 8 Specification Page	https://www.oneplus.com/8/specs
Exhibit B	Article on Warp Charge by Android Central	https://www.androidcentral.com/warp-charge
Exhibit C	Blog on Warp Charge 30 by Volta Charger	https://voltacharger.com/blogs/news/warp-charge-30-faster-charging-for-your-oneplus-device
Exhibit D	How to charge the battery correctly	https://forums.oneplus.com/threads/how-to-charge-the-battery-correctly.780695/
Exhibit E	GitHub OnePlus OSS Android Kernel Power Supply	https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L347
Exhibit F	GitHub OnePlus OSS Android Frameworks Base	https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/services/core/java/com/android/server/am/BatteryStatsService.java#L337
Exhibit G	GitHub OnePlus OSS Android Kernel APM Power	https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/apm_power.c
Exhibit H	GitHub OnePlus OSS Android Kernel Charger	https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/d0cd6cd30f8e10b209461f24462ed316c76bc913/drivers/power/supply/axp288_charger.c#L363
Exhibit I	GitHub OnePlus OSS Android Frameworks Base Battery Statistics	https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/core/java/com/android/internal/os/BatteryStatsImpl.java#L8705

Exhibit(s)	Description	Link
Exhibit J	Android Battery Statistics	https://android.googlesource.com/platform/frameworks/base.git/+/master/core/java/com/android/internal/os/BatteryStatsImpl.java
Exhibit K	OnePlus 9 Charging Test using Wrap Charger 65T	https://www.youtube.com/watch?v=CCHaMivzIvE
Exhibit L	Warp Charge 65T – Same Wattage but Better Performance	https://forums.oneplus.com/threads/deep-dive-lets-talk-charging.1403556/

Claims	**OnePlus 8, including OnePlus Smartphone based on Android OS (The accused product)**
1Pre. A method, comprising: **1a.** detecting, by an apparatus, an availability of a charger adapter;	The accused product practices a method that comprises of detecting, by an apparatus, an availability of a charger adapter. OnePlus provides smartphones such as OnePlus 8, OnePlus 7 Pro, OnePlus 8 Pro, OnePlus 8T, OnePlus 9, OnePlus 9 Pro that comes along with a OnePlus propriety charging adapter such as Warp Charge 30 Power Adapter. The Warp Charge 30 Power Adapter (i.e., charger adapter) delivers a 30-Watt (i.e., W) power output for fast charging the accused product. See Fig. 1 & Fig. 2. The accused product detects and displays the 'Warp charging' as it gets connect (i.e., available) with the Warp Charge 30 Power Adapter via the Warp type-c cable as shown in Fig. 3.

Citation 1: OnePlus 8



OnePlus 8

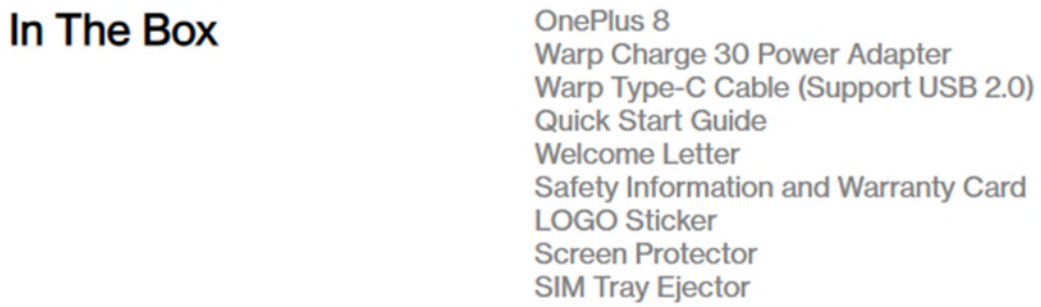
Compare with OnePlus 8 Pro

Glacial Green
Interstellar Glow
Onyx Black

Fig. 1

Source: https://www.oneplus.com/8/specs, Page 1, Last Accessed May 18, 2021, Exhibit A

Citation 2: Warp Charge 30 Power Adapter



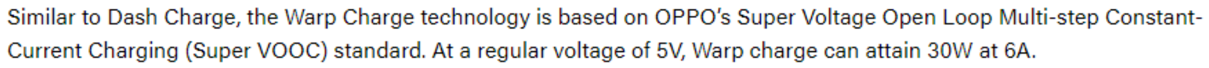
In The Box

OnePlus 8
Warp Charge 30 Power Adapter
Warp Type-C Cable (Support USB 2.0)
Quick Start Guide
Welcome Letter
Safety Information and Warranty Card
LOGO Sticker
Screen Protector
SIM Tray Ejector

Fig. 2

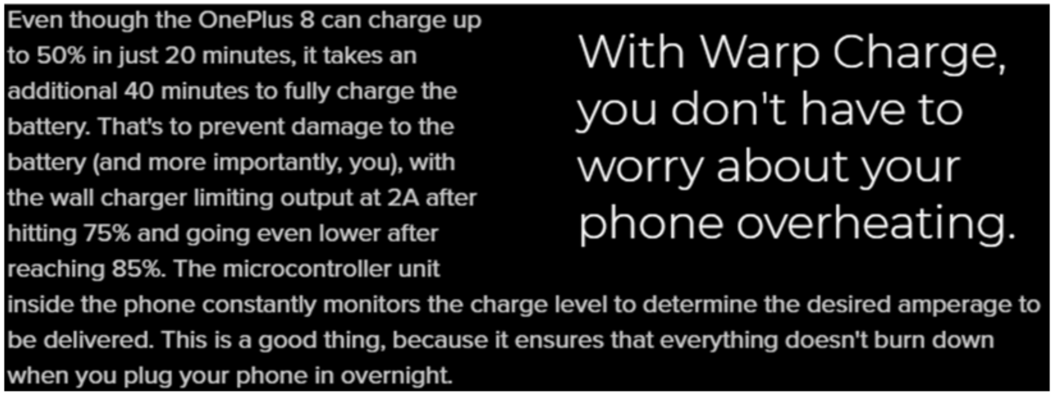
Source: https://www.oneplus.com/8/specs, Page 4, Last Accessed May 18, 2021, Exhibit A

	Citation 3: Warp Charging display Fig. 3 Source: https://www.androidcentral.com/warp-charge, Page 3, Last Accessed July 24, 2020, Exhibit B
1b. determining, by the apparatus, whether a battery charging point	The method that accused product practices comprises, whether a battery charging point is in a constant current phase or in a constant voltage phase, based on pre-determined battery charging characteristics, wherein the pre-

is in a constant current phase or in a constant voltage phase, based on pre-determined battery charging characteristics, wherein the pre-determined battery charging characteristics comprise constant current phase charging characteristics and constant voltage phase charging characteristics;	determined battery charging characteristics comprise constant current phase charging characteristics and constant voltage phase charging characteristics. As described in the patent US’708B2 [Page 14, Line 18, Column 7] “*The capacity of a battery is the quantity of electrical charge stored in the battery at a given level. The capacity of a fully charged battery is usually measured in amp-hours (AH) or milliamp-hours (mAH) and is a measure of the size of the battery.*” The Warp Charge technology uses constant current charging (i.e., constant current phase) at multi-step of voltage open loop (or, multiple stages of voltage change). See Fig. 4. **Citation 4: Warp Charge Uses Constant Current Charging**  Similar to Dash Charge, the Warp Charge technology is based on OPPO's Super Voltage Open Loop Multi-step Constant-Current Charging (Super VOOC) standard. At a regular voltage of 5V, Warp charge can attain 30W at 6A. Fig. 4 Source: https://voltacharger.com/blogs/news/warp-charge-30-faster-charging-for-your-oneplus-device, Page 2, Last Accessed May 18, 2021, Exhibit C As described in the patent US’708B2 [Page 13, Line 14, Column 5] “*As the accumulated charge in the battery 160 increases, there comes a point at which the operation of the CC/CV charger IC 154 passes from the constant current (CC) phase to the constant voltage (CV) phase. In accordance with an embodiment of the invention, estimating the remaining charging time of the rechargeable battery 160 is based on determining whether the battery’s charging point is in the constant current phase or in the constant voltage phase.*”

OnePlus 8 includes a microcontroller unit that constantly monitor the charge level to determine the desired amperage (or, current) to be delivered. The accused product comprises a charging point (e.g. integrated circuit, or another hardware) that supplies constant current/constant voltage to the battery. The accused product switches to the constant voltage phase, when the accused product (or, microcontroller unit) have change in desired amperage to prevent battery from overheating and burn down. See Fig. 5.

Citation 5: Microcontroller unit determines desired amperage



Even though the OnePlus 8 can charge up to 50% in just 20 minutes, it takes an additional 40 minutes to fully charge the battery. That's to prevent damage to the battery (and more importantly, you), with the wall charger limiting output at 2A after hitting 75% and going even lower after reaching 85%. The microcontroller unit inside the phone constantly monitors the charge level to determine the desired amperage to be delivered. This is a good thing, because it ensures that everything doesn't burn down when you plug your phone in overnight.

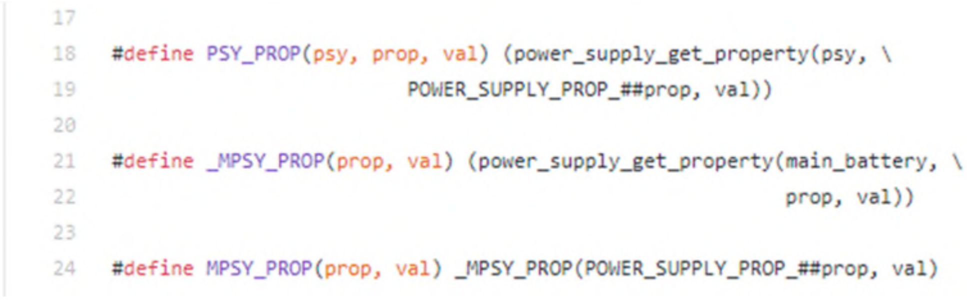
With Warp Charge, you don't have to worry about your phone overheating.

Fig. 5

Source: https://www.androidcentral.com/warp-charge, Page 3, Last Accessed July 24, 2020, Exhibit B

Fig. 6 to Fig. 9 shows a snippet of OnePlus kernel source code that is installed onto the accused product. The accused product determines whether the power supply (or, Warp Charge 30 Power Adapter) is in constant current phase or constant voltage phase and associates the psp enum value accordingly.

Citation 6: psp enum value determination



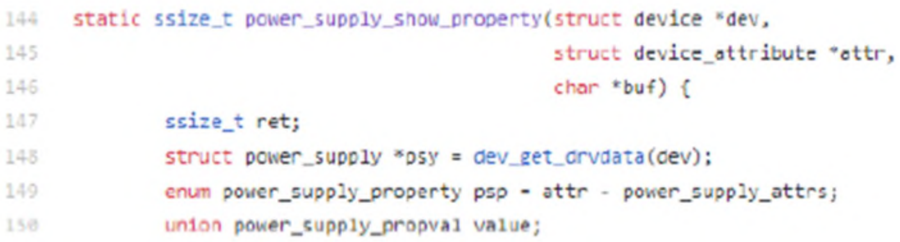
```
#define PSY_PROP(psy, prop, val) (power_supply_get_property(psy, \
                        POWER_SUPPLY_PROP_##prop, val))

#define _MPSY_PROP(prop, val) (power_supply_get_property(main_battery, \
                                                         prop, val))

#define MPSY_PROP(prop, val) _MPSY_PROP(POWER_SUPPLY_PROP_##prop, val)
```

Fig. 6

Source :

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/apm_power.c, Page 1, Last Accessed May 18, 2021, Exhibit G

Citation 7: Power Supply Attribute and Property



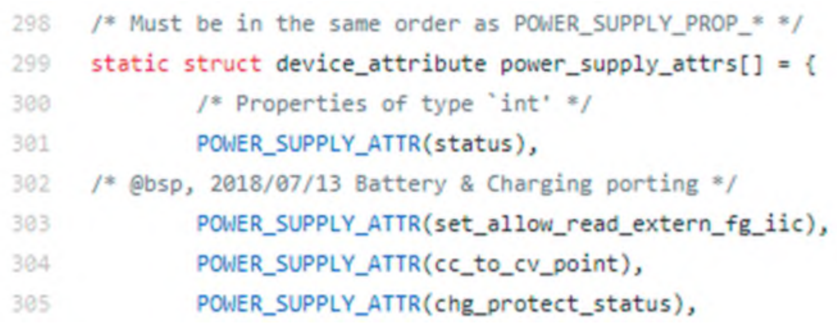
```
static ssize_t power_supply_show_property(struct device *dev,
                                          struct device_attribute *attr,
                                          char *buf) {
        ssize_t ret;
        struct power_supply *psy = dev_get_drvdata(dev);
        enum power_supply_property psp - attr - power_supply_attrs;
        union power_supply_propval value;
```

Fig. 7

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 2, Last Accessed May 18, 2021, Exhibit E

Citation 8: Device Attributes for Power Supply



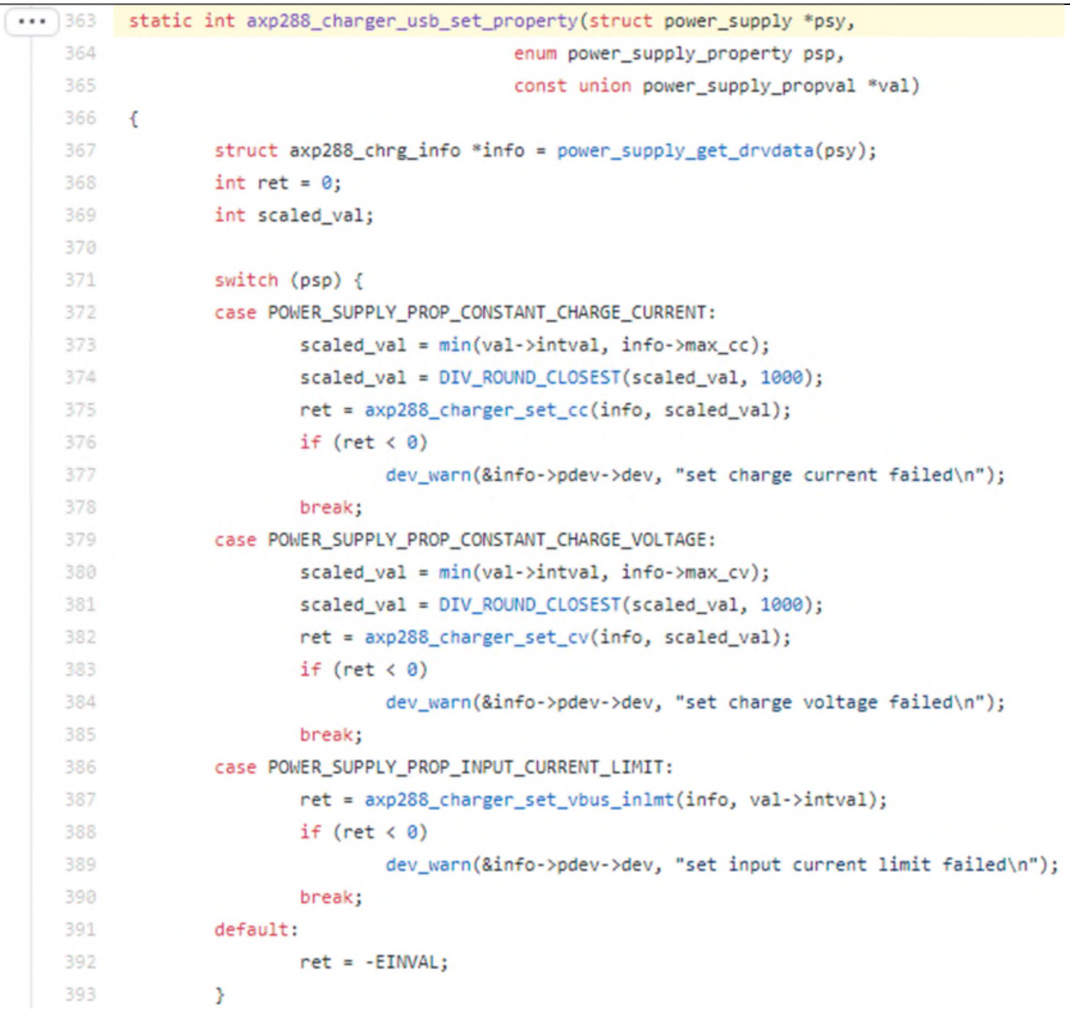
```
/* Must be in the same order as POWER_SUPPLY_PROP_* */
static struct device_attribute power_supply_attrs[] = {
        /* Properties of type `int' */
        POWER_SUPPLY_ATTR(status),
/* @bsp, 2018/07/13 Battery & Charging porting */
        POWER_SUPPLY_ATTR(set_allow_read_extern_fg_iic),
        POWER_SUPPLY_ATTR(cc_to_cv_point),
        POWER_SUPPLY_ATTR(chg_protect_status),
```

Fig. 8

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 4, Last Accessed May 18, 2021, Exhibit E

Citation 9: psp enum value determines



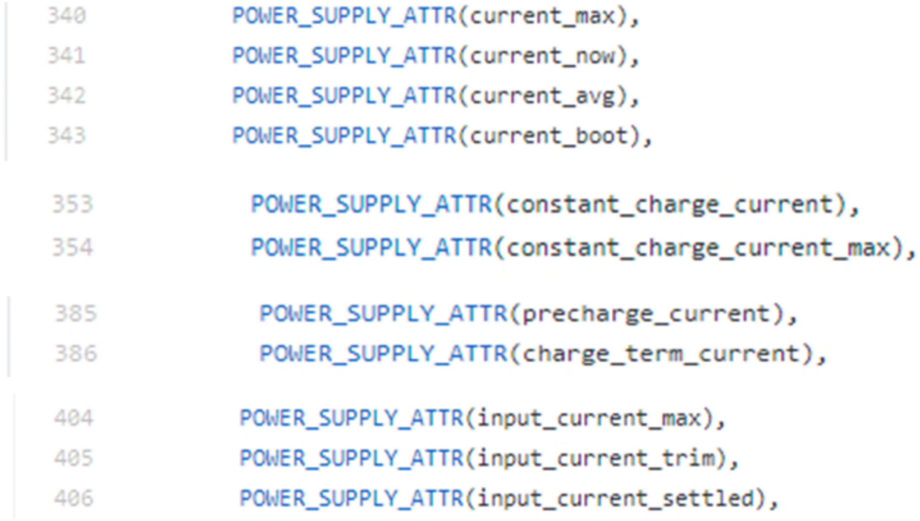
```
static int axp288_charger_usb_set_property(struct power_supply *psy,
                                    enum power_supply_property psp,
                                    const union power_supply_propval *val)
{
        struct axp288_chrg_info *info = power_supply_get_drvdata(psy);
        int ret = 0;
        int scaled_val;

        switch (psp) {
        case POWER_SUPPLY_PROP_CONSTANT_CHARGE_CURRENT:
                scaled_val = min(val->intval, info->max_cc);
                scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
                ret = axp288_charger_set_cc(info, scaled_val);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set charge current failed\n");
                break;
        case POWER_SUPPLY_PROP_CONSTANT_CHARGE_VOLTAGE:
                scaled_val = min(val->intval, info->max_cv);
                scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
                ret = axp288_charger_set_cv(info, scaled_val);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set charge voltage failed\n");
                break;
        case POWER_SUPPLY_PROP_INPUT_CURRENT_LIMIT:
                ret = axp288_charger_set_vbus_inlmt(info, val->intval);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set input current limit failed\n");
                break;
        default:
                ret = -EINVAL;
        }
```

Fig. 9

Source:https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/d0cd6cd30f8e10b209461f24462ed316c76bc913/drivers/power/supply/axp288_charger.c#L363, Page 5, Last Accessed May 18, 2021, Exhibit H

Fig. 10 and Fig. 11 shows snippets of OnePlus kernel source code that is installed onto the accused product. The accused product stores current and voltage attributes which are used to compute the constant current phase charging characteristics and constant voltage phase charging characteristics, respectively.

Citation 10: Constant Current Phase Charging Attributes



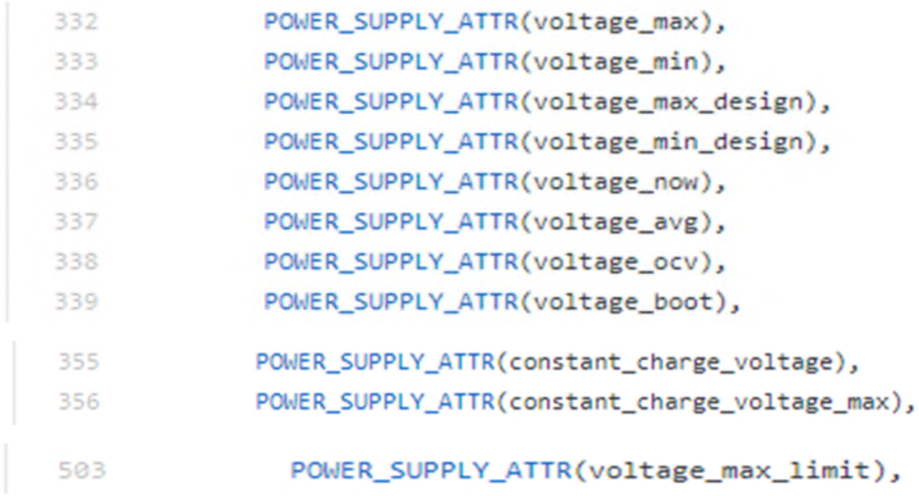
```
        POWER_SUPPLY_ATTR(current_max),
        POWER_SUPPLY_ATTR(current_now),
        POWER_SUPPLY_ATTR(current_avg),
        POWER_SUPPLY_ATTR(current_boot),

        POWER_SUPPLY_ATTR(constant_charge_current),
        POWER_SUPPLY_ATTR(constant_charge_current_max),

        POWER_SUPPLY_ATTR(precharge_current),
        POWER_SUPPLY_ATTR(charge_term_current),

        POWER_SUPPLY_ATTR(input_current_max),
        POWER_SUPPLY_ATTR(input_current_trim),
        POWER_SUPPLY_ATTR(input_current_settled),
```

Fig. 10

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 5, Last Accessed May 18 2021, Exhibit E

Citation 11: Constant Voltage Phase Charging Attributes



```
        POWER_SUPPLY_ATTR(voltage_max),
        POWER_SUPPLY_ATTR(voltage_min),
        POWER_SUPPLY_ATTR(voltage_max_design),
        POWER_SUPPLY_ATTR(voltage_min_design),
        POWER_SUPPLY_ATTR(voltage_now),
        POWER_SUPPLY_ATTR(voltage_avg),
        POWER_SUPPLY_ATTR(voltage_ocv),
        POWER_SUPPLY_ATTR(voltage_boot),

        POWER_SUPPLY_ATTR(constant_charge_voltage),
        POWER_SUPPLY_ATTR(constant_charge_voltage_max),

        POWER_SUPPLY_ATTR(voltage_max_limit),
```

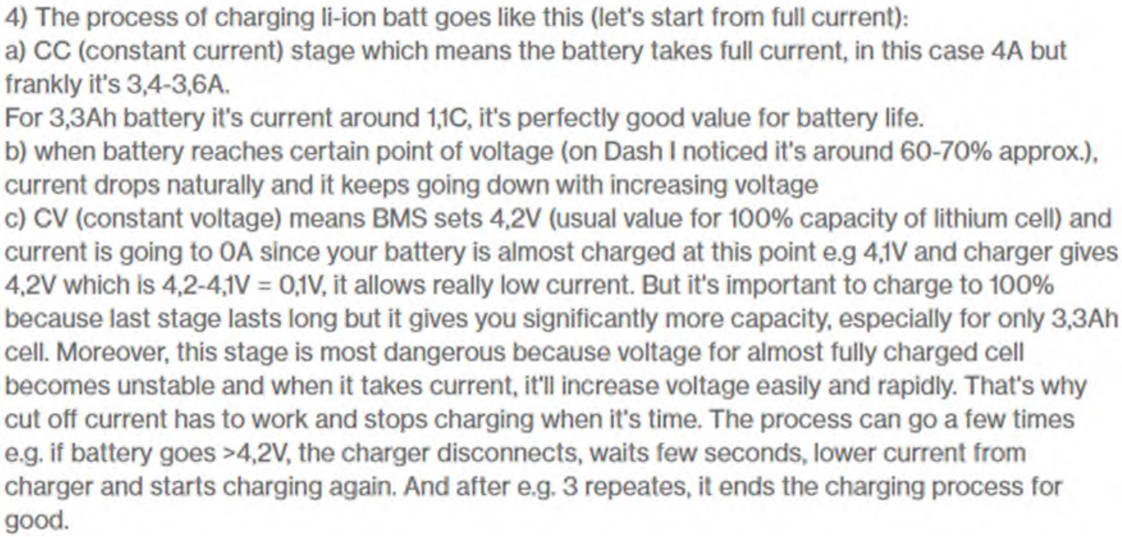
Fig. 11

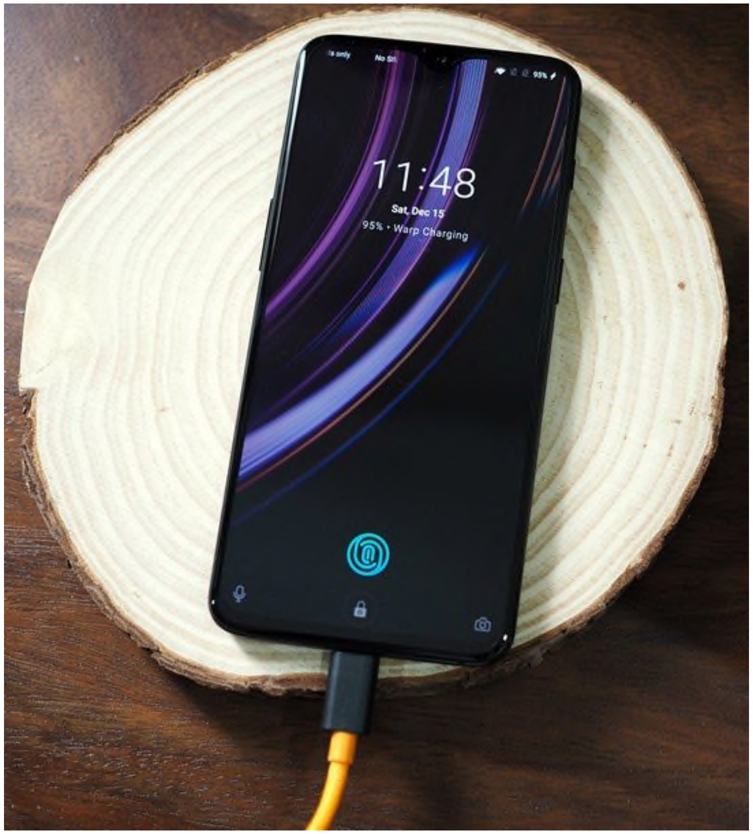
Source: https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 4, 5 & 6, Last Accessed May 18, 2021, Exhibit E

The accused product uses various characteristics such as battery percentage upon the time of charging, the time period over which the battery is charging, battery voltage to determine that if the charging point is in the constant current phase or the constant voltage phase. See Fig. 12 and Fig. 13.

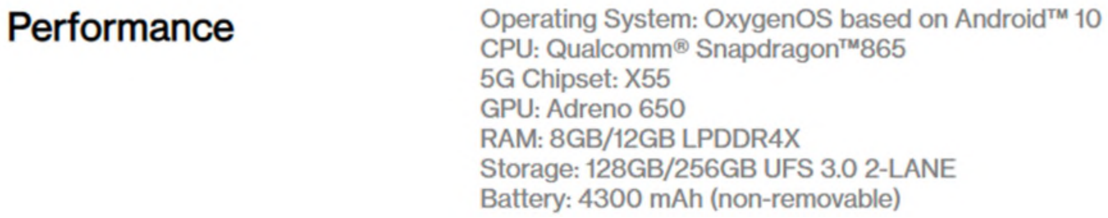
Citation 12: Battery percentage on OnePlus 8

	   Fig. 12 Source: https://www.androidcentral.com/warp-charge, Page 3, Last Accessed July 24, 2020, Exhibit B

	Citation 13: Constant Current and Constant Voltage Phase  4) The process of charging li-ion batt goes like this (let's start from full current): a) CC (constant current) stage which means the battery takes full current, in this case 4A but frankly it's 3,4-3,6A. For 3,3Ah battery it's current around 1,1C, it's perfectly good value for battery life. b) when battery reaches certain point of voltage (on Dash I noticed it's around 60-70% approx.), current drops naturally and it keeps going down with increasing voltage c) CV (constant voltage) means BMS sets 4,2V (usual value for 100% capacity of lithium cell) and current is going to 0A since your battery is almost charged at this point e.g 4,1V and charger gives 4,2V which is 4,2-4,1V = 0,1V, it allows really low current. But it's important to charge to 100% because last stage lasts long but it gives you significantly more capacity, especially for only 3,3Ah cell. Moreover, this stage is most dangerous because voltage for almost fully charged cell becomes unstable and when it takes current, it'll increase voltage easily and rapidly. That's why cut off current has to work and stops charging when it's time. The process can go a few times e.g. if battery goes >4,2V, the charger disconnects, waits few seconds, lower current from charger and starts charging again. And after e.g. 3 repeates, it ends the charging process for good. Fig. 13 Source: https://forums.oneplus.com/threads/how-to-charge-the-battery-correctly.780695/, Page 3, Last Accessed May 18, 2021, Exhibit D
1c. calculating, by the apparatus, a time remaining to charge in the constant current phase based on the constant current phase charging characteristics, if the	The method that accused product practices comprises, calculating a time remaining to charge in the constant current phase based on the constant current phase charging characteristics, if the battery charging point is in the constant current phase, and calculating a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant voltage phase. The accused product detects and displays the type of power adapter that charges the device. The accused product uses the type of charging point connected and various other pre-determined characteristics such as the amount of electrical charge stores in the battery (i.e., battery level), full capacity (i.e., 4300mAH) of the battery installed in

battery charging point is in the constant current phase; and **1d.** calculating, by the apparatus, a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant voltage phase;	the smartphone, and more to estimate the time remaining to charge the smartphone's battery in the near real-time. See Fig. 14 & Fig. 15. **Citation 14: Warp Charging Displays on OnePlus 8**  Fig. 14 Source: https://www.androidcentral.com/warp-charge, Page 3, Last Accessed July 24, 2020, Exhibit B

Citation 15: Non-removable 4300 mAh battery capacity



Performance

Operating System: OxygenOS based on Android™ 10
CPU: Qualcomm® Snapdragon™865
5G Chipset: X55
GPU: Adreno 650
RAM: 8GB/12GB LPDDR4X
Storage: 128GB/256GB UFS 3.0 2-LANE
Battery: 4300 mAh (non-removable)

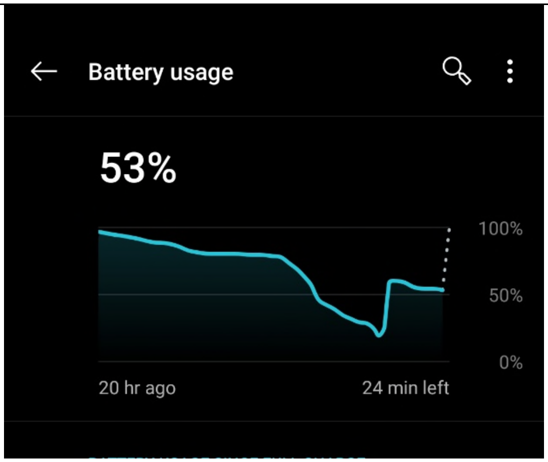
Fig. 15

Source: https://www.oneplus.com/8/specs, Page 1-2, Last Accessed May 18, 2021, Exhibit A

As described in the patent US'708B2 [Page 15, Line 57, Column 10] "*Step 408: if the battery charging point is in the constant current phase, calculating a time remaining to charge in the constant current phase based on the constant current phase charging characteristics and a time remaining to charge in the constant Voltage phase based on the constant Voltage phase charging characteristics. In step 408 the present capacity and the present charging current are also inputs for this calculation.*" and [Page 15, Line 65, Column 10] "*Step 410: if the battery charging point is in the constant Voltage phase, calculating a time remaining to charge in the constant Voltage phase based on the constant Voltage phase charging characteristics. In step 410 the present charging current is also an input for this calculation.*"

The accused product (e.g., OnePlus 3T) calculates the time to charge the device to 100% battery level. See Fig. 16.

Citation 16: Estimation of time left to reach the 100% battery level



Battery usage

53%

Fig. 16

Source: Snippet taken from OnePlus 3T

The Fig. 17, shows the estimate of remaining charging time on lock screen of the accused product (e.g, OnePlus 9).

Citation 17: OnePlus 9 lock screen display remaining charging time estimate

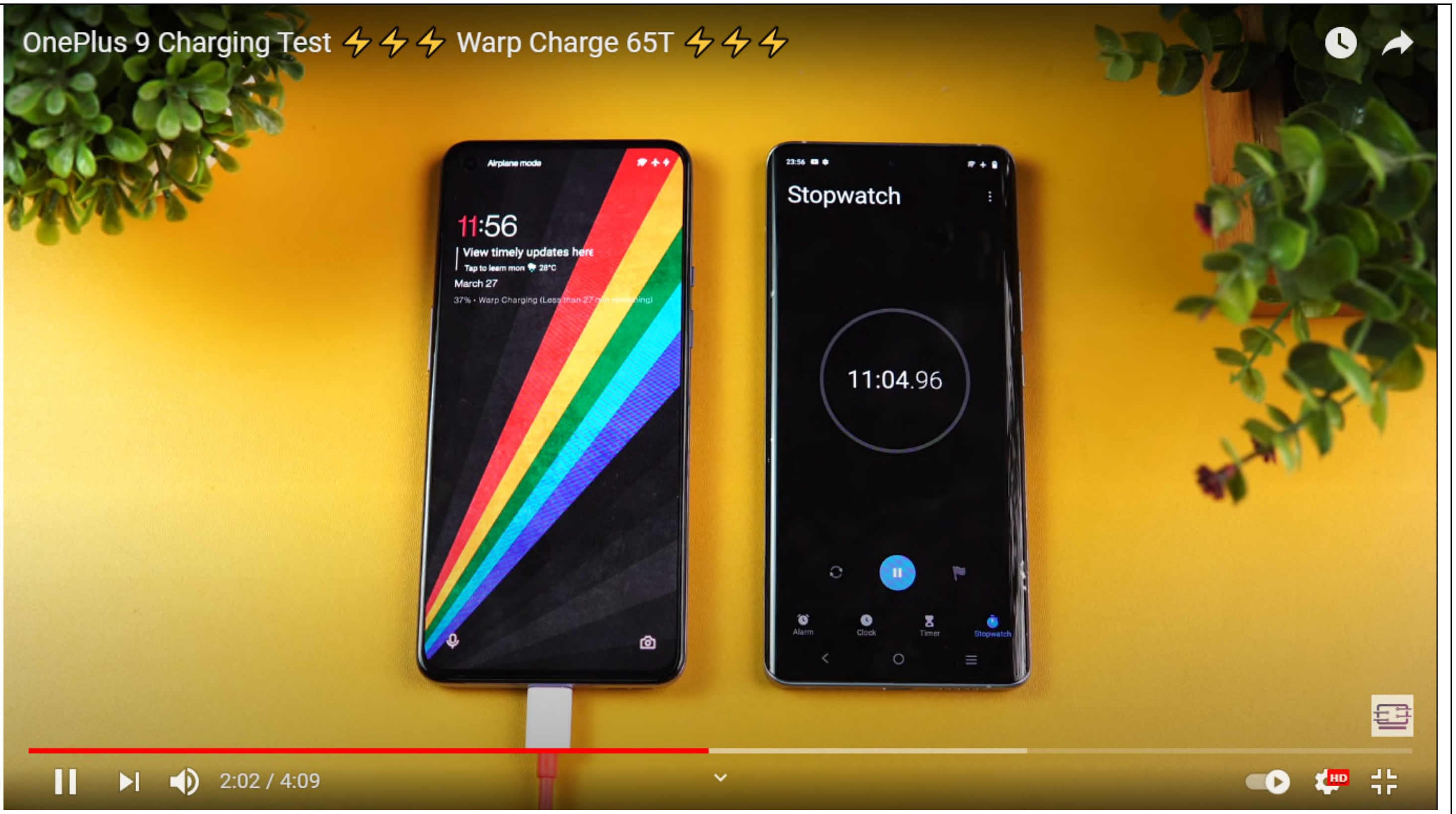
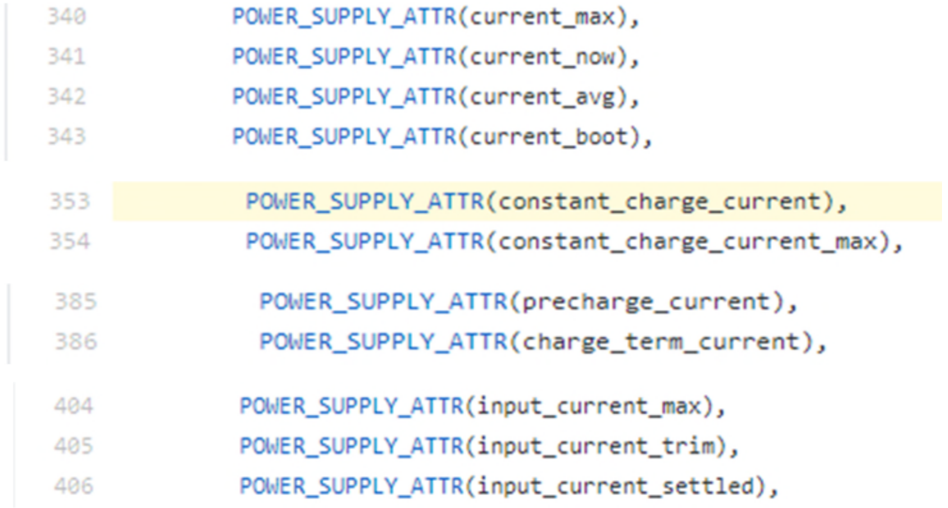


Fig. 17

Source: https://www.youtube.com/watch?v=CCHaMivzIvE, Time- 2:02, Last accessed June 24, 2021, Exhibit K

The accused product determines various characteristics (i.e. constant voltage phase charging characteristics) such as voltage that the charging point feeds, battery level, total battery capacity, and more such parameters at the constant current phase. Moreover, the accused product also determines various characteristics (i.e. constant current phase charging characteristics) such as current that the charging point feeds, battery level, total battery capacity, and more at the constant voltage phase. See Fig. 18 to **Error! Reference source not found.**.

Citation 18: Constant Current Phase Charging Attributes



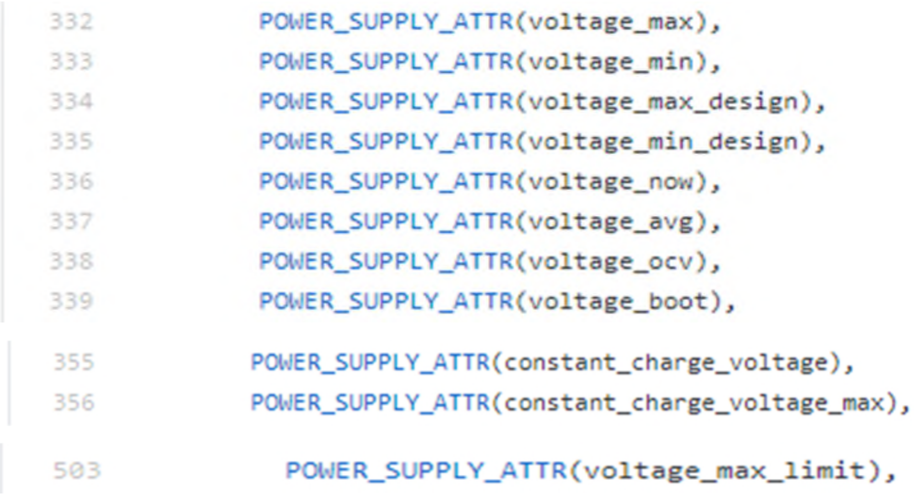
```
340            POWER_SUPPLY_ATTR(current_max),
341            POWER_SUPPLY_ATTR(current_now),
342            POWER_SUPPLY_ATTR(current_avg),
343            POWER_SUPPLY_ATTR(current_boot),

353            POWER_SUPPLY_ATTR(constant_charge_current),
354            POWER_SUPPLY_ATTR(constant_charge_current_max),

385            POWER_SUPPLY_ATTR(precharge_current),
386            POWER_SUPPLY_ATTR(charge_term_current),

404            POWER_SUPPLY_ATTR(input_current_max),
405            POWER_SUPPLY_ATTR(input_current_trim),
406            POWER_SUPPLY_ATTR(input_current_settled),
```

Fig. 18

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 5, Last Accessed May 18, 2021, Exhibit E

Citation 19: Constant Voltage Phase Charging Attributes



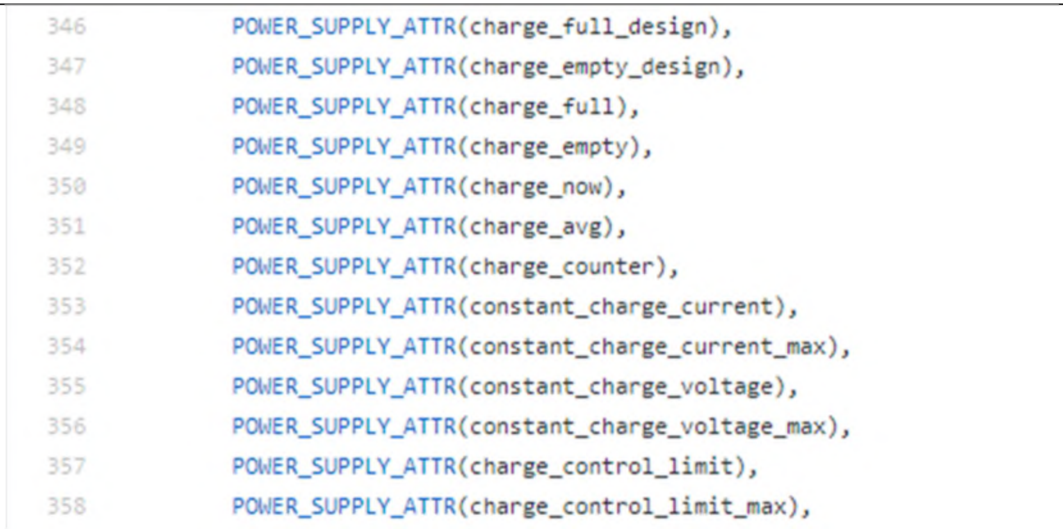
```
        POWER_SUPPLY_ATTR(voltage_max),
        POWER_SUPPLY_ATTR(voltage_min),
        POWER_SUPPLY_ATTR(voltage_max_design),
        POWER_SUPPLY_ATTR(voltage_min_design),
        POWER_SUPPLY_ATTR(voltage_now),
        POWER_SUPPLY_ATTR(voltage_avg),
        POWER_SUPPLY_ATTR(voltage_ocv),
        POWER_SUPPLY_ATTR(voltage_boot),

        POWER_SUPPLY_ATTR(constant_charge_voltage),
        POWER_SUPPLY_ATTR(constant_charge_voltage_max),

        POWER_SUPPLY_ATTR(voltage_max_limit),
```

Fig. 19

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 4, 5 & 6, Last Accessed May 18, 2021, Exhibit E

Citation 20: Stored Charge and Current Charge



```
	POWER_SUPPLY_ATTR(charge_full_design),
	POWER_SUPPLY_ATTR(charge_empty_design),
	POWER_SUPPLY_ATTR(charge_full),
	POWER_SUPPLY_ATTR(charge_empty),
	POWER_SUPPLY_ATTR(charge_now),
	POWER_SUPPLY_ATTR(charge_avg),
	POWER_SUPPLY_ATTR(charge_counter),
	POWER_SUPPLY_ATTR(constant_charge_current),
	POWER_SUPPLY_ATTR(constant_charge_current_max),
	POWER_SUPPLY_ATTR(constant_charge_voltage),
	POWER_SUPPLY_ATTR(constant_charge_voltage_max),
	POWER_SUPPLY_ATTR(charge_control_limit),
	POWER_SUPPLY_ATTR(charge_control_limit_max),
```

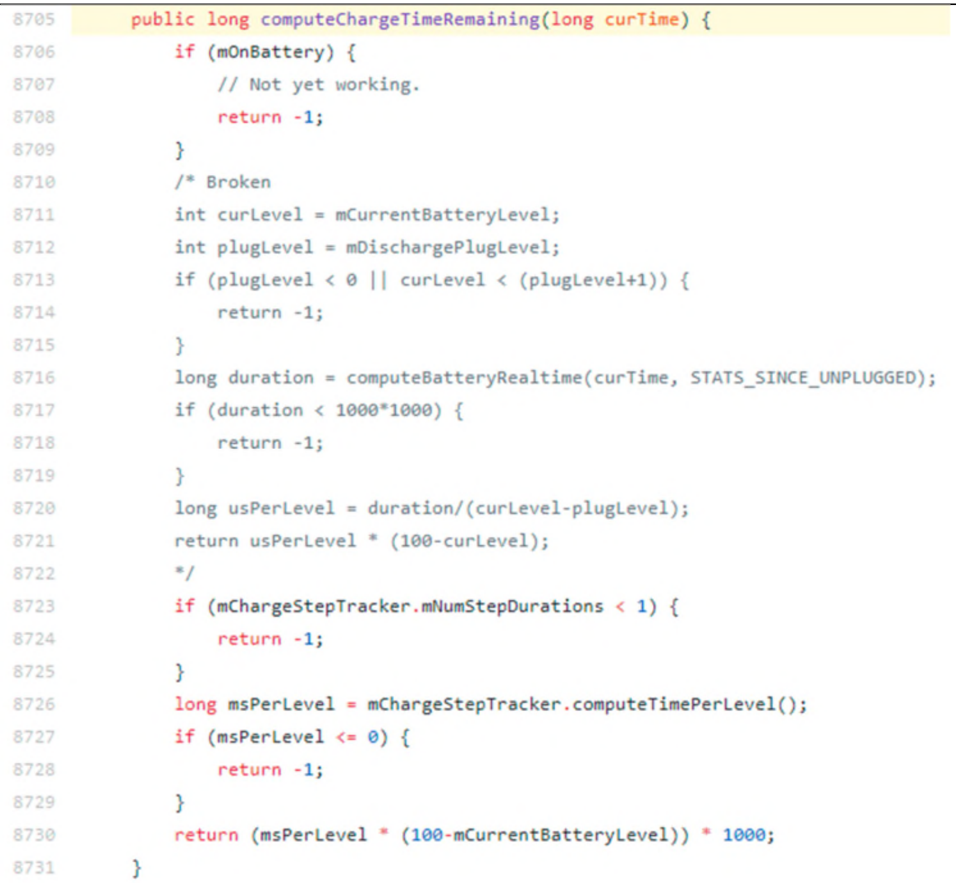
Fig. 20

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 5, Last Accessed May 18, 2021, Exhibit E

As an example, shown in Fig. 21 is a code snippet that shows the calculation of charging time estimate. Upon information and belief, the accused product uses 'mChargeStepTracker' to estimate (or, calculate) the time remaining to charge the battery, based on constant current/voltage characteristics. See Fig. 22.

Citation 21: Battery Charging and Discharging Attributes

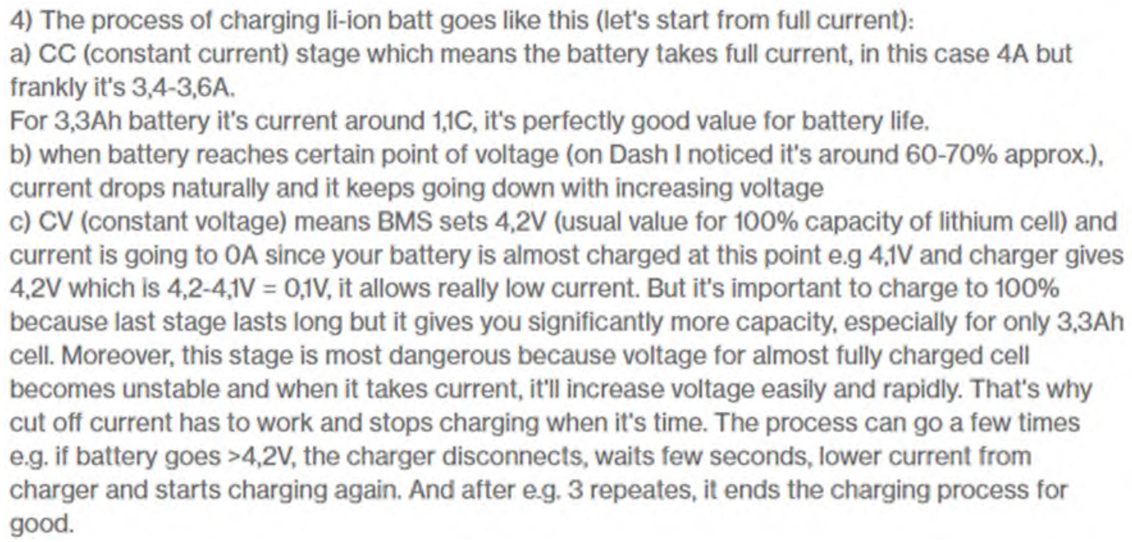


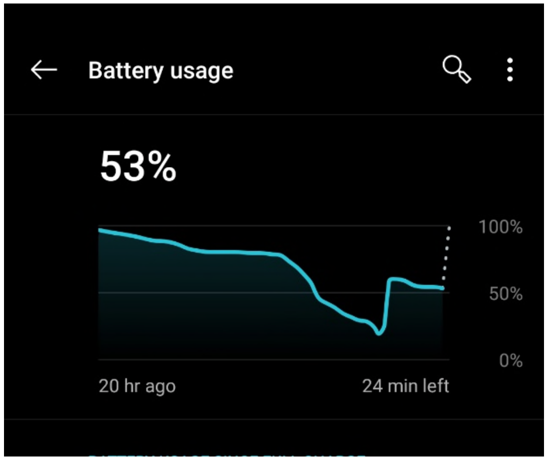
```
    public long computeChargeTimeRemaining(long curTime) {
        if (mOnBattery) {
            // Not yet working.
            return -1;
        }
        /* Broken
        int curLevel = mCurrentBatteryLevel;
        int plugLevel = mDischargePlugLevel;
        if (plugLevel < 0 || curLevel < (plugLevel+1)) {
            return -1;
        }
        long duration = computeBatteryRealtime(curTime, STATS_SINCE_UNPLUGGED);
        if (duration < 1000*1000) {
            return -1;
        }
        long usPerLevel = duration/(curLevel-plugLevel);
        return usPerLevel * (100-curLevel);
        */
        if (mChargeStepTracker.mNumStepDurations < 1) {
            return -1;
        }
        long msPerLevel = mChargeStepTracker.computeTimePerLevel();
        if (msPerLevel <= 0) {
            return -1;
        }
        return (msPerLevel * (100-mCurrentBatteryLevel)) * 1000;
    }
```

Fig. 21

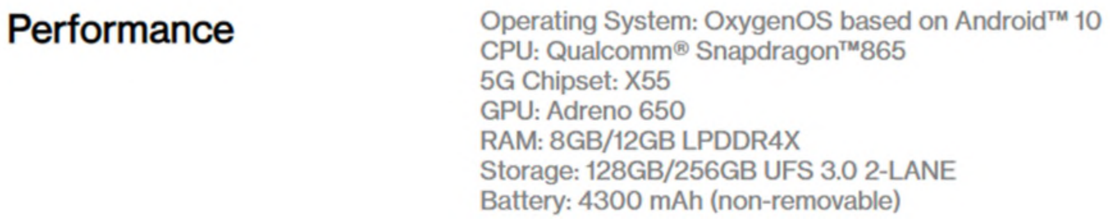
Source:

https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/core/java/com/android/internal/os/BatteryStatsImpl.java#L8705, Page 243-244, Last Accessed May 18, 2021, Exhibit I

	Citation 22: Constant Current Phase and Constant Voltage Phase  4) The process of charging li-ion batt goes like this (let's start from full current): a) CC (constant current) stage which means the battery takes full current, in this case 4A but frankly it's 3,4-3,6A. For 3,3Ah battery it's current around 1,1C, it's perfectly good value for battery life. b) when battery reaches certain point of voltage (on Dash I noticed it's around 60-70% approx.), current drops naturally and it keeps going down with increasing voltage c) CV (constant voltage) means BMS sets 4,2V (usual value for 100% capacity of lithium cell) and current is going to 0A since your battery is almost charged at this point e.g 4,1V and charger gives 4,2V which is 4,2-4,1V = 0,1V, it allows really low current. But it's important to charge to 100% because last stage lasts long but it gives you significantly more capacity, especially for only 3,3Ah cell. Moreover, this stage is most dangerous because voltage for almost fully charged cell becomes unstable and when it takes current, it'll increase voltage easily and rapidly. That's why cut off current has to work and stops charging when it's time. The process can go a few times e.g. if battery goes >4,2V, the charger disconnects, waits few seconds, lower current from charger and starts charging again. And after e.g. 3 repeates, it ends the charging process for good. Fig. 22 Source: https://forums.oneplus.com/threads/how-to-charge-the-battery-correctly.780695/, Page 3, Last Accessed May 18, 2021, Exhibit D
1e. wherein the time remaining to charge in the constant current phase is based on stored charge characteristics in the constant current phase, when the battery	The accused product practices a method, wherein the time remaining to charge in the constant current phase is based on stored charge characteristics in the constant current phase, when the battery charging point is in the constant current phase and wherein battery stored charge characteristics comprises a battery stored charge value based on monitored tracking of battery charging and discharging. As described in the patent US'708B2 [Page 14, Line 48, Column 7] "*Along with the measured data for datasets 210 and 215, the following information may be stored in the device 100 as dataset 220 for each combination of*

charging point is in the constant current phase and wherein battery stored charge characteristics comprises a battery stored charge value based on monitored tracking of battery charging and discharging.	*device 100 battery 160 type: 1 Charge current during CC phase (IBATCC); 2) Battery 160 voltage when the charging switches to the CV phase (VBATCV). If the battery 160 voltage slightly increases during the CV phase, a value of the lowest Voltage in the constant Voltage phase may be chosen; and 3Duration of the CV phase (TCVTOT).”* The accused product calculates and displays the amount of time life to charge the battery to 100% charge level as shown in See Fig. 23 **Citation 23: Estimate time left to reach the 100% battery level**  Battery usage 53% 100% 50% 0% 20 hr ago 24 min left Fig. 23 Source: Snippet taken from OnePlus 3T OnePlus 8 is deployed with Android based Operating System (i.e., OS). See Fig. 24.

Citation 24: OnePlus 8 runs Android based OS



Performance

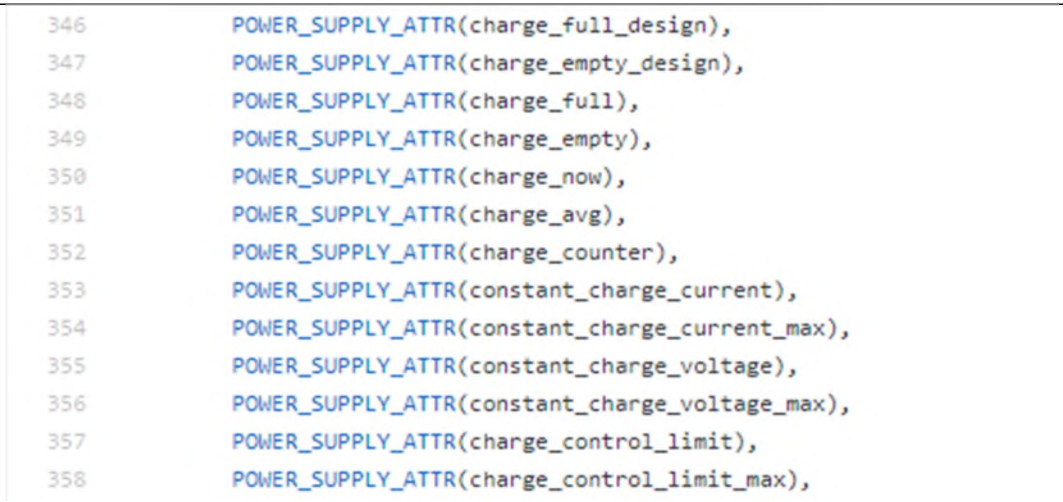
Operating System: OxygenOS based on Android™ 10
CPU: Qualcomm® Snapdragon™865
5G Chipset: X55
GPU: Adreno 650
RAM: 8GB/12GB LPDDR4X
Storage: 128GB/256GB UFS 3.0 2-LANE
Battery: 4300 mAh (non-removable)

Fig. 24

Source: https://www.oneplus.com/8/specs, Page 1-2, Last Accessed May 18, 2021, Exhibit A

Fig. 25 and Fig. 26 show snippets of OnePlus kernel source code that is installed onto the accused product, and the accused product stores battery stored charge value, which is based on the attributes like “mCurrentBatteryLevel” and “mDischargePlugLevel” (i.e., battery charging and discharging).

Citation 25: Stored Charge and Current Charge



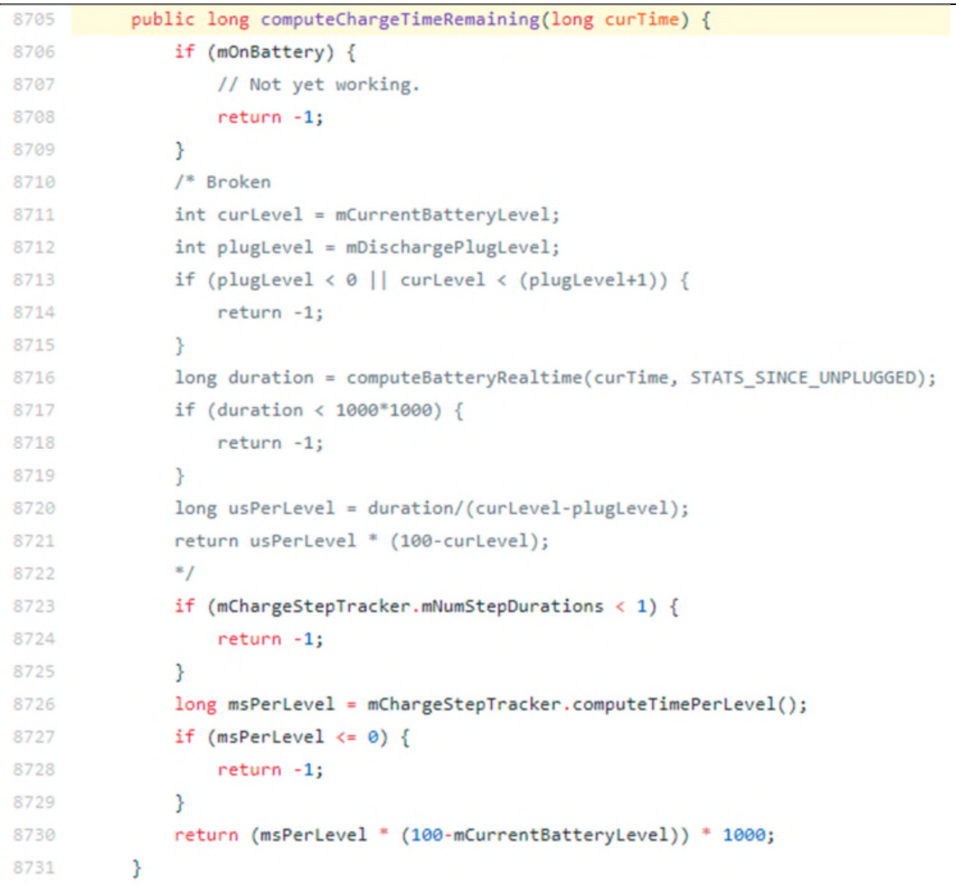
```
	POWER_SUPPLY_ATTR(charge_full_design),
	POWER_SUPPLY_ATTR(charge_empty_design),
	POWER_SUPPLY_ATTR(charge_full),
	POWER_SUPPLY_ATTR(charge_empty),
	POWER_SUPPLY_ATTR(charge_now),
	POWER_SUPPLY_ATTR(charge_avg),
	POWER_SUPPLY_ATTR(charge_counter),
	POWER_SUPPLY_ATTR(constant_charge_current),
	POWER_SUPPLY_ATTR(constant_charge_current_max),
	POWER_SUPPLY_ATTR(constant_charge_voltage),
	POWER_SUPPLY_ATTR(constant_charge_voltage_max),
	POWER_SUPPLY_ATTR(charge_control_limit),
	POWER_SUPPLY_ATTR(charge_control_limit_max),
```

Fig. 25

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 5, Last Accessed May 18, 2021, Exhibit E

Citation 26: Battery Charging and Discharging Attributes



```java
    public long computeChargeTimeRemaining(long curTime) {
        if (mOnBattery) {
            // Not yet working.
            return -1;
        }
        /* Broken
        int curLevel = mCurrentBatteryLevel;
        int plugLevel = mDischargePlugLevel;
        if (plugLevel < 0 || curLevel < (plugLevel+1)) {
            return -1;
        }
        long duration = computeBatteryRealtime(curTime, STATS_SINCE_UNPLUGGED);
        if (duration < 1000*1000) {
            return -1;
        }
        long usPerLevel = duration/(curLevel-plugLevel);
        return usPerLevel * (100-curLevel);
        */
        if (mChargeStepTracker.mNumStepDurations < 1) {
            return -1;
        }
        long msPerLevel = mChargeStepTracker.computeTimePerLevel();
        if (msPerLevel <= 0) {
            return -1;
        }
        return (msPerLevel * (100-mCurrentBatteryLevel)) * 1000;
    }
```

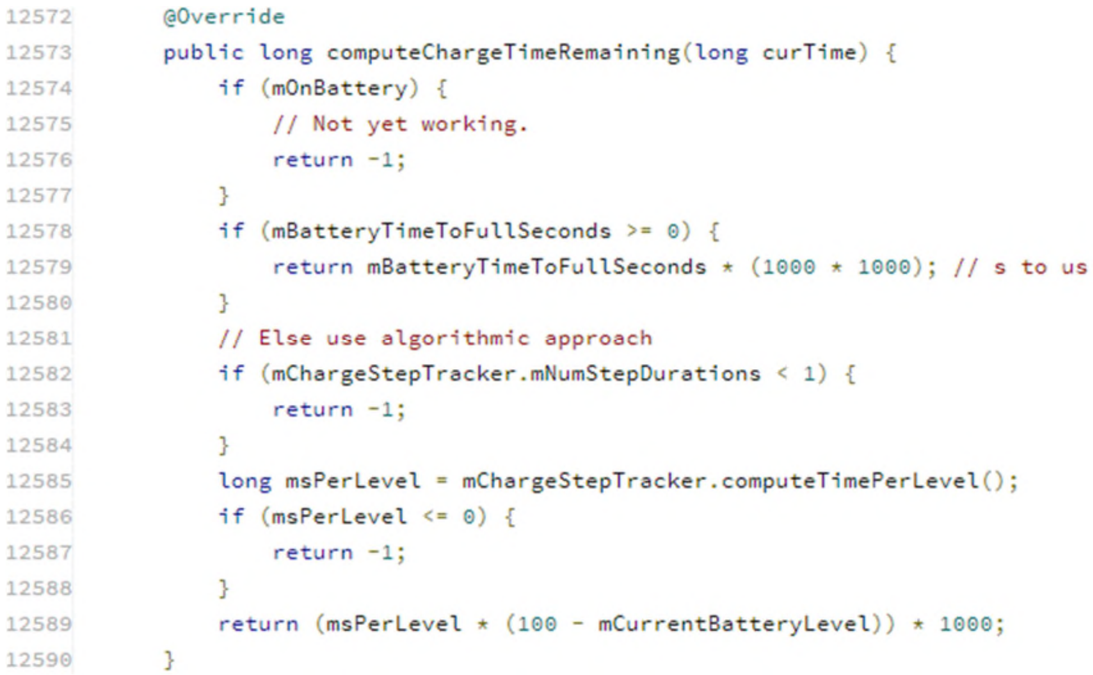
Fig. 26

Source:

https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/core/java/com/android/internal/os/BatteryStatsImpl.java#L8705, Page 243-244, Last Accessed May 18, 2021, Exhibit I

	Fig. 27 shows the snippet of source code of Android that computes and stores the time in millisecond (i.e. ms) requires for charging the battery by 1%, in ‘msPerLevel’, which depends on both charging and discharging. As an example, the time to charge 1% depends on the charging of the battery, as with increased power supply there is a decrease in the displayed charging time. Also, the time to charge 1% depends on the discharging of the battery caused by the processing load, resulting in an increase in the battery's charging time. The accused product running android-based OS computes the time remaining to charge battery, in constant current, by multiplying the ‘msPerLevel’ with battery level remaining to charge. See Fig. 27 - Fig. 30.

Citation 27: Computation of remaining time based on per step level



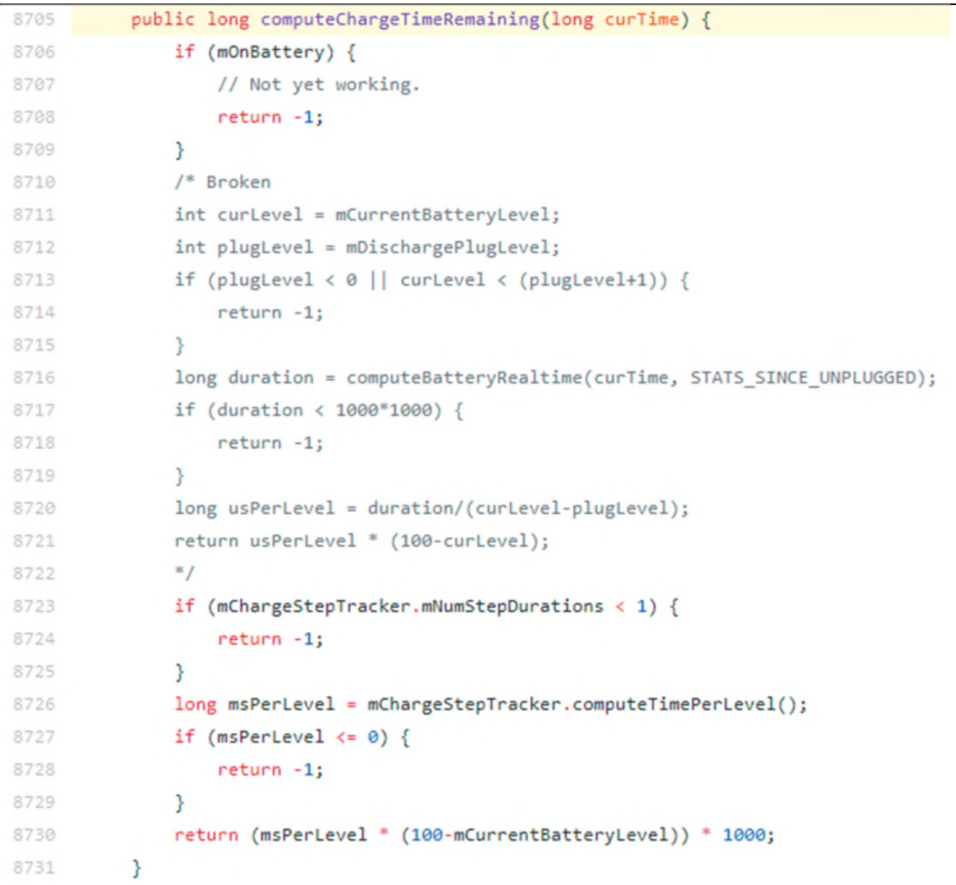
```
    @Override
    public long computeChargeTimeRemaining(long curTime) {
        if (mOnBattery) {
            // Not yet working.
            return -1;
        }
        if (mBatteryTimeToFullSeconds >= 0) {
            return mBatteryTimeToFullSeconds * (1000 * 1000); // s to us
        }
        // Else use algorithmic approach
        if (mChargeStepTracker.mNumStepDurations < 1) {
            return -1;
        }
        long msPerLevel = mChargeStepTracker.computeTimePerLevel();
        if (msPerLevel <= 0) {
            return -1;
        }
        return (msPerLevel * (100 - mCurrentBatteryLevel)) * 1000;
    }
```

Fig. 27

Source:

https://android.googlesource.com/platform/frameworks/base.git/+/master/core/java/com/android/internal/os/BatteryStatsImpl.java#12604, Page 1, Last Accessed May 18, 2021, Exhibit J

Citation 28: Battery Charging and Discharging Attributes



```
    public long computeChargeTimeRemaining(long curTime) {
        if (mOnBattery) {
            // Not yet working.
            return -1;
        }
        /* Broken
        int curLevel = mCurrentBatteryLevel;
        int plugLevel = mDischargePlugLevel;
        if (plugLevel < 0 || curLevel < (plugLevel+1)) {
            return -1;
        }
        long duration = computeBatteryRealtime(curTime, STATS_SINCE_UNPLUGGED);
        if (duration < 1000*1000) {
            return -1;
        }
        long usPerLevel = duration/(curLevel-plugLevel);
        return usPerLevel * (100-curLevel);
        */
        if (mChargeStepTracker.mNumStepDurations < 1) {
            return -1;
        }
        long msPerLevel = mChargeStepTracker.computeTimePerLevel();
        if (msPerLevel <= 0) {
            return -1;
        }
        return (msPerLevel * (100-mCurrentBatteryLevel)) * 1000;
    }
```

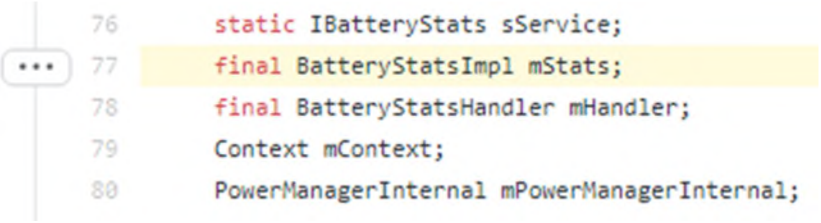
Fig. 28

Source:

https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/core/java/com/android/internal/os/BatteryStatsImpl.java#L8705, Page 243-244, Last Accessed May 18, 2021

Exhibit I

Citation 29: BatteryStats.Java uses BatteryStatsImpl data to provide



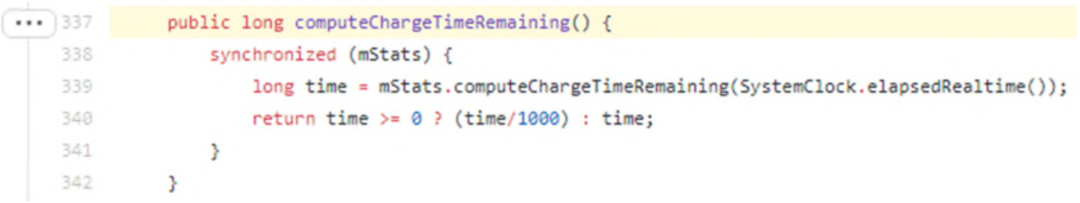
```
    static IBatteryStats sService;
    final BatteryStatsImpl mStats;
    final BatteryStatsHandler mHandler;
    Context mContext;
    PowerManagerInternal mPowerManagerInternal;
```

Fig. 29

Source:

https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/services/core/java/com/android/server/am/BatteryStatsService.java#L337, Page 3, Last Accessed May 18, 2021, Exhibit F

Citation 30: BatteryStats.Java uses BatteryStatsImpl data to provide



```
    public long computeChargeTimeRemaining() {
        synchronized (mStats) {
            long time = mStats.computeChargeTimeRemaining(SystemClock.elapsedRealtime());
            return time >= 0 ? (time/1000) : time;
        }
    }
```

Fig. 30

Source:

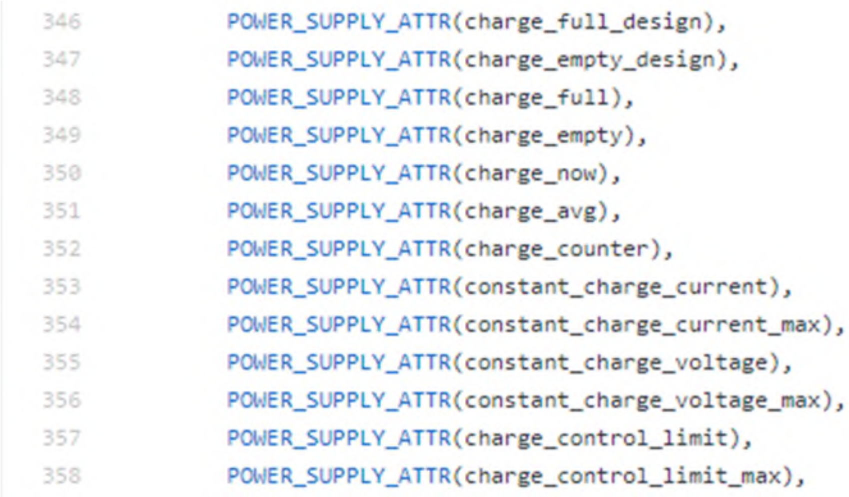
https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb93

<table>
<tr>
<td></td>
<td>2/services/core/java/com/android/server/am/BatteryStatsService.java#L337, Page 9, Last Accessed May 18, 2021, Exhibit F</td>
</tr>
<tr>
<td>2. The method of claim 1, wherein the battery stored charge characteristics comprise data that provides a remaining charging time estimate based on a battery stored charge estimate.</td>
<td>

The accused product practices a method, wherein the battery stored charge characteristics comprise data that provides a remaining charging time estimate based on a battery stored charge estimate.

Fig. 31 shows snippets of the OnePlus kernel source code that is installed onto the accused product, and the accused product stores battery stored charge value and accordingly "computeChargeTimeRemaining" (i.e., remaining charging time) is estimated. See Fig. 32.

Citation 31: Stored Charge and Current Charge



```
POWER_SUPPLY_ATTR(charge_full_design),
POWER_SUPPLY_ATTR(charge_empty_design),
POWER_SUPPLY_ATTR(charge_full),
POWER_SUPPLY_ATTR(charge_empty),
POWER_SUPPLY_ATTR(charge_now),
POWER_SUPPLY_ATTR(charge_avg),
POWER_SUPPLY_ATTR(charge_counter),
POWER_SUPPLY_ATTR(constant_charge_current),
POWER_SUPPLY_ATTR(constant_charge_current_max),
POWER_SUPPLY_ATTR(constant_charge_voltage),
POWER_SUPPLY_ATTR(constant_charge_voltage_max),
POWER_SUPPLY_ATTR(charge_control_limit),
POWER_SUPPLY_ATTR(charge_control_limit_max),
```

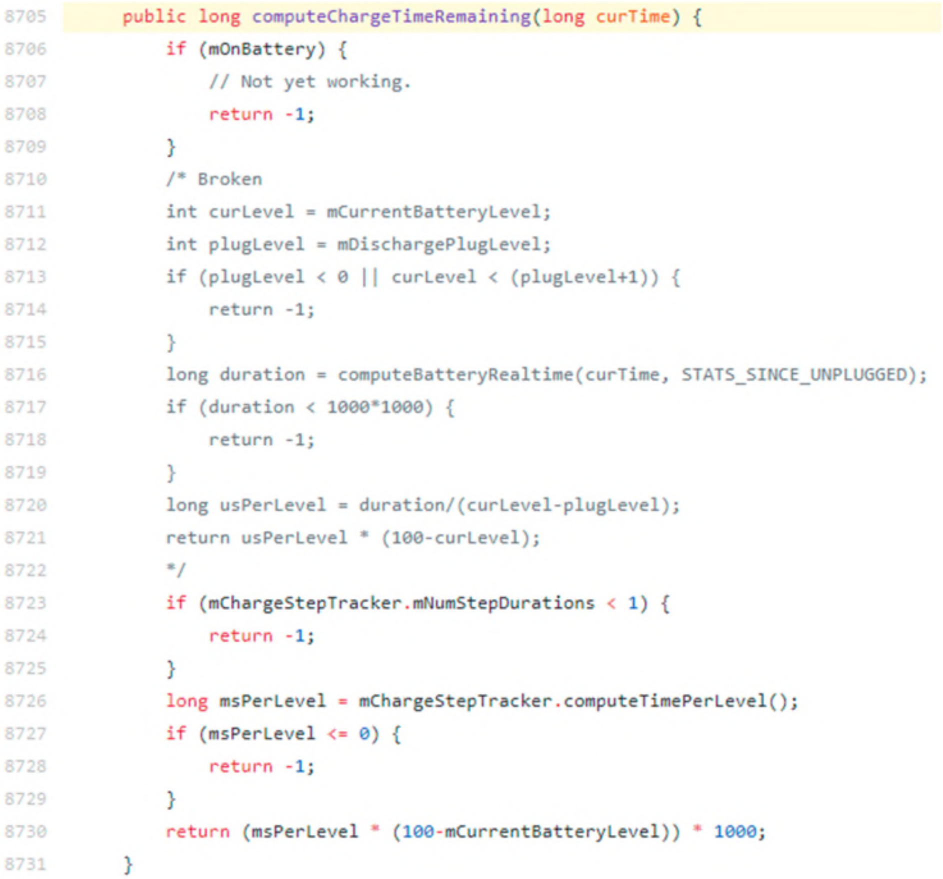
Fig. 31

</td>
</tr>
</table>

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 5, Last Accessed May 18, 2021, Exhibit E

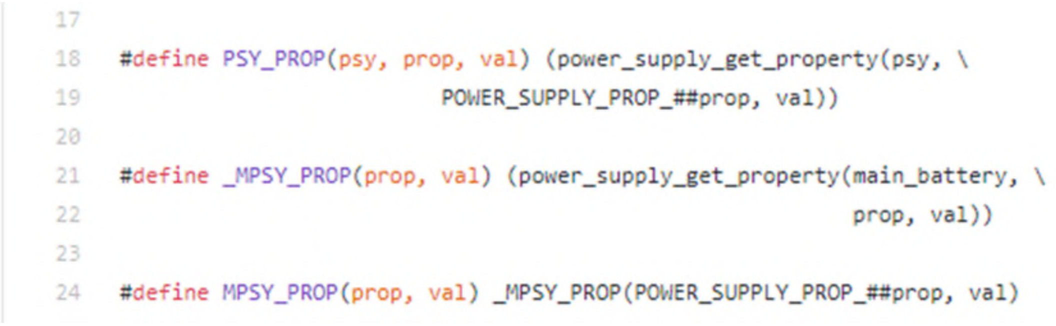
Citation 32: Remaining Charging Time Estimation



```
    public long computeChargeTimeRemaining(long curTime) {
        if (mOnBattery) {
            // Not yet working.
            return -1;
        }
        /* Broken
        int curLevel = mCurrentBatteryLevel;
        int plugLevel = mDischargePlugLevel;
        if (plugLevel < 0 || curLevel < (plugLevel+1)) {
            return -1;
        }
        long duration = computeBatteryRealtime(curTime, STATS_SINCE_UNPLUGGED);
        if (duration < 1000*1000) {
            return -1;
        }
        long usPerLevel = duration/(curLevel-plugLevel);
        return usPerLevel * (100-curLevel);
        */
        if (mChargeStepTracker.mNumStepDurations < 1) {
            return -1;
        }
        long msPerLevel = mChargeStepTracker.computeTimePerLevel();
        if (msPerLevel <= 0) {
            return -1;
        }
        return (msPerLevel * (100-mCurrentBatteryLevel)) * 1000;
    }
```

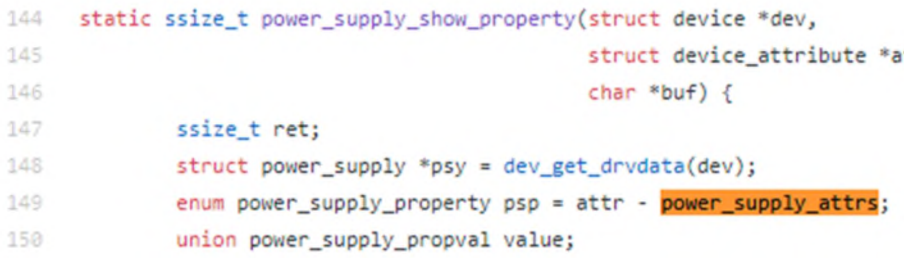
Fig. 32

	Source: https://github.com/OnePlusOSS/android_frameworks_base/blob/95ba353daa05fdd1c6011554ab7152fadfcdb932/core/java/com/android/internal/os/BatteryStatsImpl.java#L8705, Page 243-244, Last Accessed May 18, 2021, Exhibit I

4. The method of claim 1, wherein the time remaining to charge in the constant voltage phase is based on charging current characteristics in the constant voltage phase, when the battery charging point is in the constant voltage phase.	The accused product practices a method, wherein the time remaining to charge in the constant voltage phase is based on charging current characteristics in the constant voltage phase, when the battery charging point is in the constant voltage phase.
	Fig. 33 to Fig. 36 shows snippets of OnePlus kernel source code that is installed onto the accused product. The accused product determines whether the power supply (or, Warp Charge 30 Power Adapter) is in constant current phase or constant voltage phase by using 'power_supply_property psp' which comprises attributes for the battery charging characteristics and associates the psp enum value accordingly.
	Citation 33: psp enum value determination
	
	`#define PSY_PROP(psy, prop, val) (power_supply_get_property(psy, \`
	`                POWER_SUPPLY_PROP_##prop, val))`
	
	`#define _MPSY_PROP(prop, val) (power_supply_get_property(main_battery, \`
	`                            prop, val))`
	
	`#define MPSY_PROP(prop, val) _MPSY_PROP(POWER_SUPPLY_PROP_##prop, val)`
	Fig. 33

Source: https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/apm_power.c, Page 1, Last Accessed May 18, 2021, Exhibit G

Citation 34: Power Supply Property

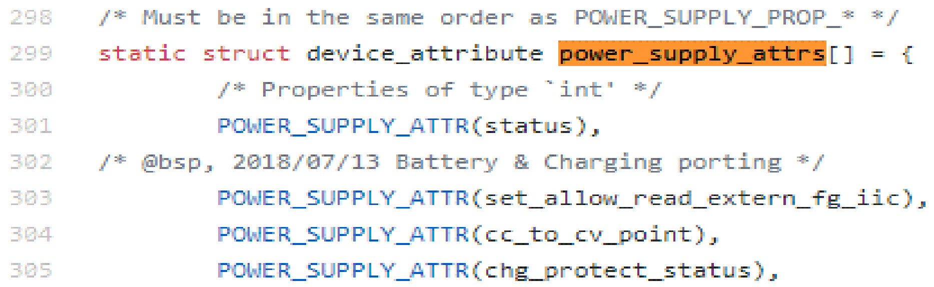


```
static ssize_t power_supply_show_property(struct device *dev,
                                          struct device_attribute *attr,
                                          char *buf) {
        ssize_t ret;
        struct power_supply *psy = dev_get_drvdata(dev);
        enum power_supply_property psp = attr - power_supply_attrs;
        union power_supply_propval value;
```

Fig. 34

Source: https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 2, Last Accessed May 18, 2021, Exhibit E

Citation 35: Device Attributes for Power Supply



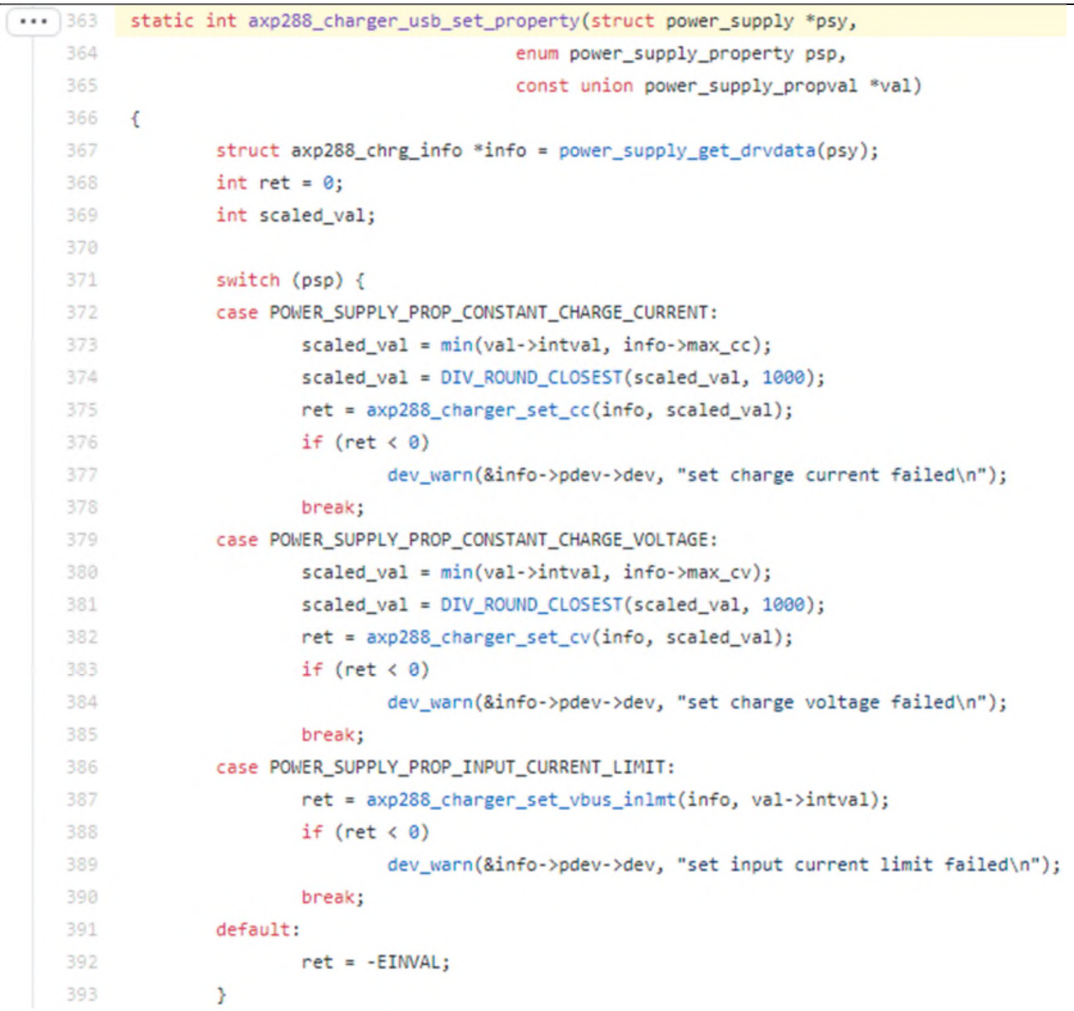
```
/* Must be in the same order as POWER_SUPPLY_PROP_* */
static struct device_attribute power_supply_attrs[] = {
        /* Properties of type `int' */
        POWER_SUPPLY_ATTR(status),
/* @bsp, 2018/07/13 Battery & Charging porting */
        POWER_SUPPLY_ATTR(set_allow_read_extern_fg_iic),
        POWER_SUPPLY_ATTR(cc_to_cv_point),
        POWER_SUPPLY_ATTR(chg_protect_status),
```

Fig. 35

Source:

https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 4, Last Accessed May 18, 2021, Exhibit E

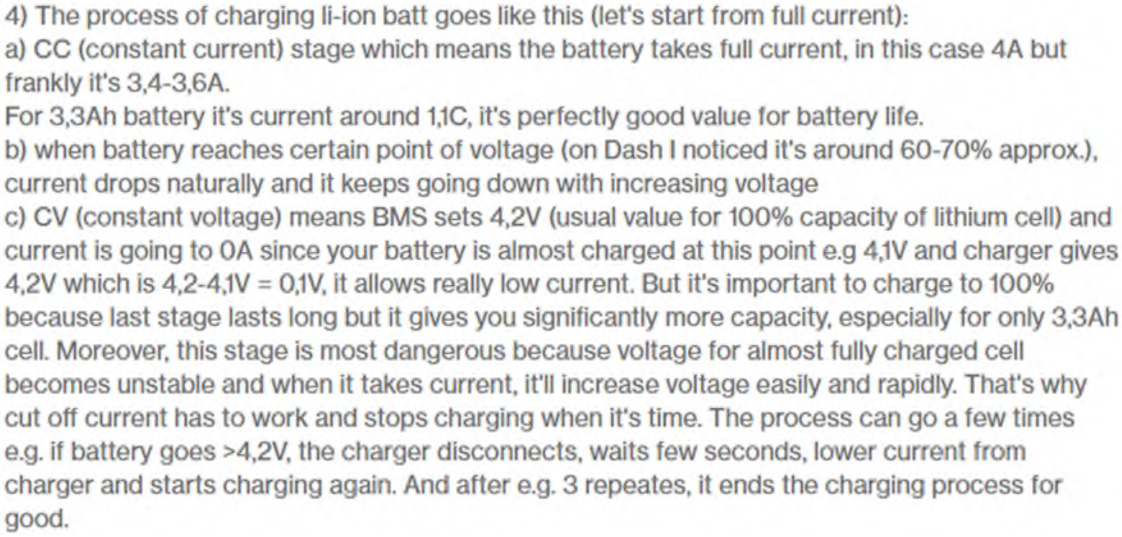
Citation 36: Constant Current & Constant Voltage in Power Supply Property

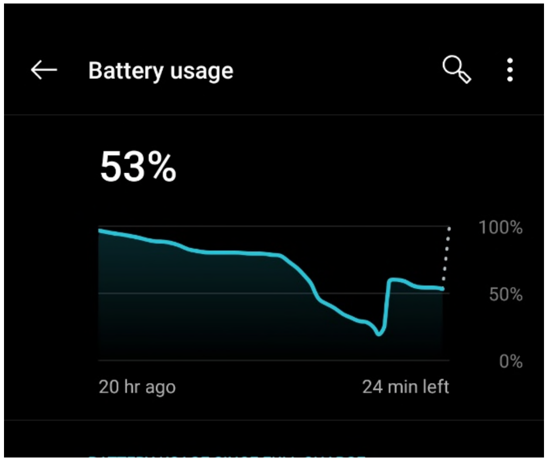


```
static int axp288_charger_usb_set_property(struct power_supply *psy,
                                    enum power_supply_property psp,
                                    const union power_supply_propval *val)
{
        struct axp288_chrg_info *info = power_supply_get_drvdata(psy);
        int ret = 0;
        int scaled_val;

        switch (psp) {
        case POWER_SUPPLY_PROP_CONSTANT_CHARGE_CURRENT:
                scaled_val = min(val->intval, info->max_cc);
                scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
                ret = axp288_charger_set_cc(info, scaled_val);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set charge current failed\n");
                break;
        case POWER_SUPPLY_PROP_CONSTANT_CHARGE_VOLTAGE:
                scaled_val = min(val->intval, info->max_cv);
                scaled_val = DIV_ROUND_CLOSEST(scaled_val, 1000);
                ret = axp288_charger_set_cv(info, scaled_val);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set charge voltage failed\n");
                break;
        case POWER_SUPPLY_PROP_INPUT_CURRENT_LIMIT:
                ret = axp288_charger_set_vbus_inlmt(info, val->intval);
                if (ret < 0)
                        dev_warn(&info->pdev->dev, "set input current limit failed\n");
                break;
        default:
                ret = -EINVAL;
        }
```

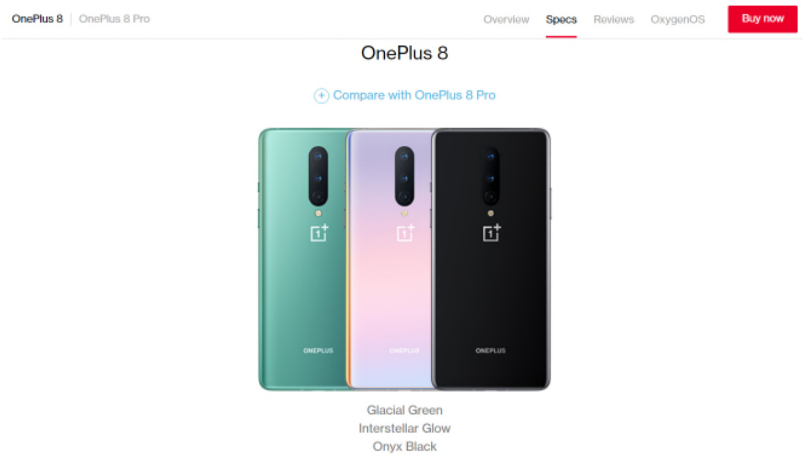
Fig. 36

Source:https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/d0cd6cd30f8e10b209461f24462ed316c76bc913/drivers/power/supply/axp288_charger.c#L363, Page 5, Last Accessed May 18, 2021, Exhibit H

	When the accused product is connected to the charger, firstly it takes full current, but when the battery reaches a certain point of voltage, the type of charging switches to constant voltage from constant current. In the CV (i.e., constant voltage) mode, the voltage is set to 4.2V, and accordingly current (i.e., charging current characteristics) is allowed to charge device to 100%. See Fig. 37. **Citation 37: Constant Current Phase and Constant Voltage Phase**  4) The process of charging li-ion batt goes like this (let's start from full current): a) CC (constant current) stage which means the battery takes full current, in this case 4A but frankly it's 3,4-3,6A. For 3,3Ah battery it's current around 1,1C, it's perfectly good value for battery life. b) when battery reaches certain point of voltage (on Dash I noticed it's around 60-70% approx.), current drops naturally and it keeps going down with increasing voltage c) CV (constant voltage) means BMS sets 4,2V (usual value for 100% capacity of lithium cell) and current is going to 0A since your battery is almost charged at this point e.g 4,1V and charger gives 4,2V which is 4,2-4,1V = 0,1V, it allows really low current. But it's important to charge to 100% because last stage lasts long but it gives you significantly more capacity, especially for only 3,3Ah cell. Moreover, this stage is most dangerous because voltage for almost fully charged cell becomes unstable and when it takes current, it'll increase voltage easily and rapidly. That's why cut off current has to work and stops charging when it's time. The process can go a few times e.g. if battery goes >4,2V, the charger disconnects, waits few seconds, lower current from charger and starts charging again. And after e.g. 3 repeates, it ends the charging process for good. Fig. 37 Source: https://forums.oneplus.com/threads/how-to-charge-the-battery-correctly.780695/, Page 3, Last Accessed May 18, 2021, Exhibit D

	Citation 38: Estimation of time left to reach the 100% battery level  Battery usage 53% 100% 50% 0% 20 hr ago 24 min left Fig. 38 Source: Snippet taken from OnePlus 3T
5Pre. The method of claim 1, further comprising: **5a.** identifying, by the apparatus, a correct category of the charger adapter after detecting its availability; and	The accused product practices a method comprising identifying, by the apparatus, a correct category of the charger adapter after detecting its availability. OnePlus provides smartphones such as OnePlus 8 that come along with a Warp Charge 30 Power Adapter. The Warp Charge 30 Power Adapter (i.e., charger adapter) delivers a 30-Watt power output for fast charging the accused product. See Fig. 39 and Fig. 40.

Citation 39: OnePlus 8



OnePlus 8

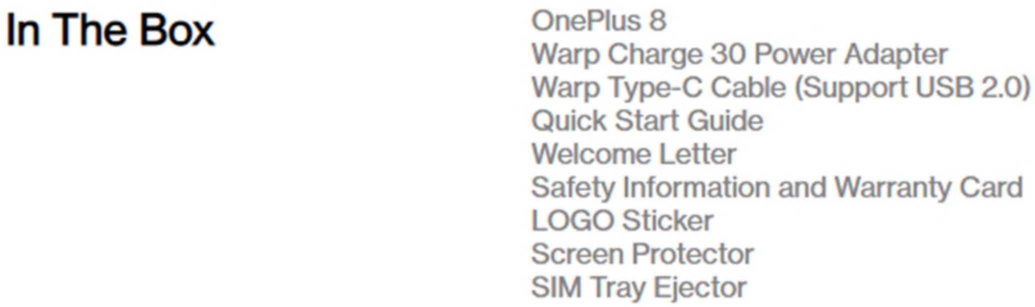
Compare with OnePlus 8 Pro

Glacial Green
Interstellar Glow
Onyx Black

Fig. 39

Source: https://www.oneplus.com/8/specs, Page 1, Last Accessed May 18, 2021, Exhibit A

Citation 40: Warp Charge 30 Power Adapter



In The Box

OnePlus 8
Warp Charge 30 Power Adapter
Warp Type-C Cable (Support USB 2.0)
Quick Start Guide
Welcome Letter
Safety Information and Warranty Card
LOGO Sticker
Screen Protector
SIM Tray Ejector

Fig. 40

Source: https://www.oneplus.com/8/specs, Page 4, Last Accessed May 18, 2021, Exhibit A

The accused product displays “Warp charging” (i.e., correct category of the charger adapter) as it gets connected (i.e., detecting its availability) with the Warp Charge 30 Power Adapter via the Warp Type-C cable as shown in Fig. 41.

Citation 41: Warp Charging display

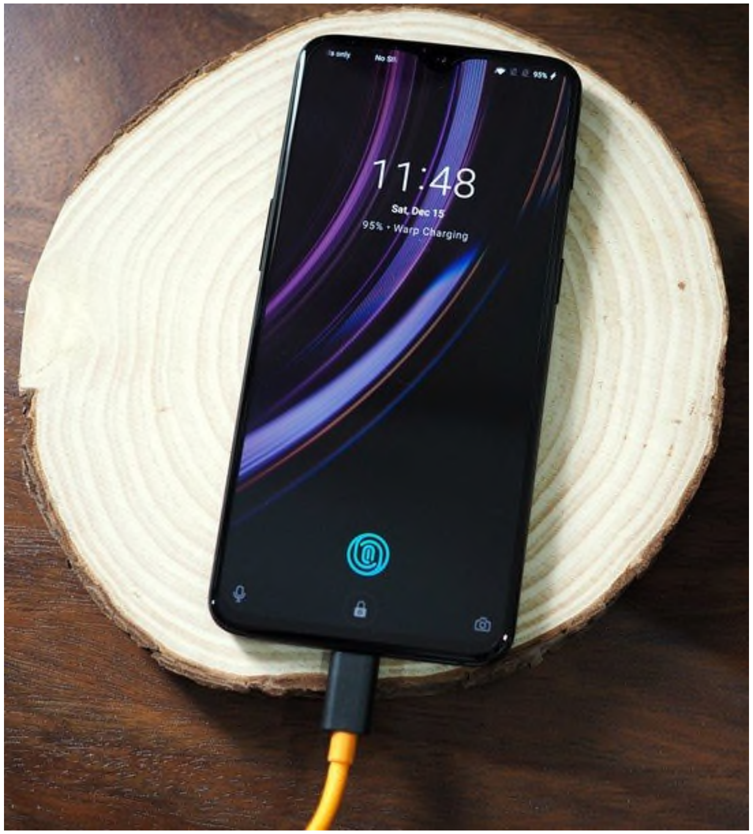


Fig. 41

Source: https://www.androidcentral.com/warp-charge, Page 3, Last Accessed July 24, 2020, Exhibit B

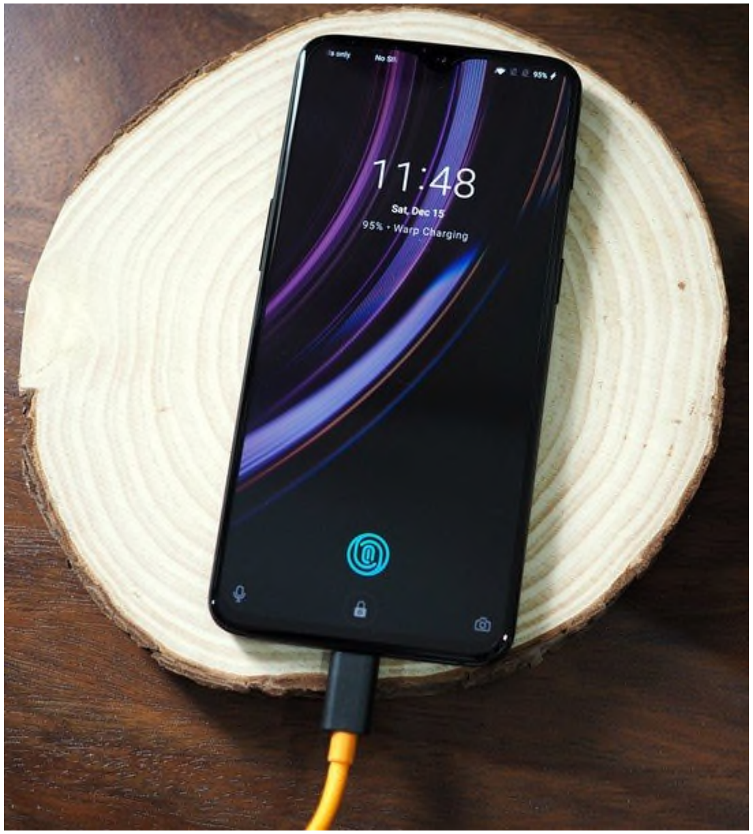
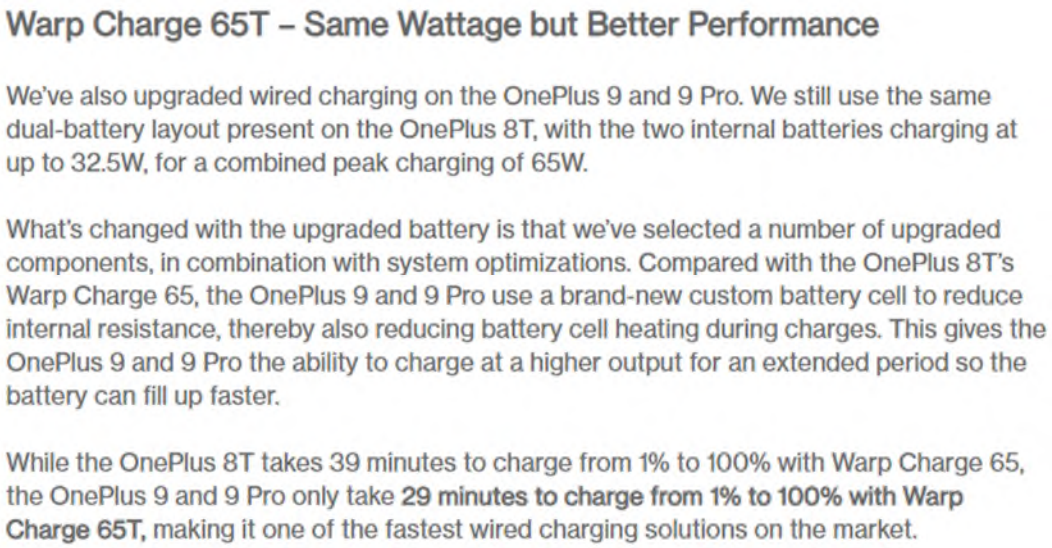
5b. configuring, by the apparatus, battery charging based in the category of the charger adapter.	The accused product practices a method comprising configuring, by the apparatus, battery charging based in the category of the charger adapter. Once the accused product is connected with the Warp Charge 30 Power Adapter (i.e., charger adapter), the battery charging is configured, and the current battery status is displayed on the screen. See Fig. 42. **Citation 42: Warp Charging display** 

	Fig. 42 Source: https://www.androidcentral.com/warp-charge, Page 3, Last Accessed July 24, 2020, Exhibit B
6Pre. The method of claim 1, further comprising: **6a.** identifying, by the apparatus, a correct category of the charger adapter after detecting its availability; and	The accused product practices a method identifying, by the apparatus, a correct category of the charger adapter after detecting its availability. *Refer to supporting evidence of claim element 5[a].*
6b. using, by the apparatus, the category of the charger adapter to improve accuracy of an initial remaining charging time estimation.	The accused product practices a method comprising using, by the apparatus, the category of the charger adapter to improve accuracy of an initial remaining charging time estimation. In an exemplary scenario, when a Warp Charge 65T (e.g., OnePlus charger adapter provided inbox with the device) is used to charge the OnePlus 9, the OnePlus 9 determines the adapter information (e.g., manufacturer, power details, etc.) to calculate and display the remaining charging time.

In an exemplary scenario, when a general adapter (e.g., not OnePlus adapter) with same power as used in previous scenario (i.e., same power as Warp Charge 65T) is used to charge the OnePlus 9, the OnePlus 9 determines the adapter information (e.g., manufacturer, power details, etc.) to calculate and display the remaining charging time.

Upon information and belief, the estimation of remaining charging time in both the cases are different due to the type of adapter taken into consideration for estimation. See Fig. 43 and Fig. 44.

Citation 43: Warp Charge 65T require 29 Minutes to fully charge OnePlus 9



Warp Charge 65T – Same Wattage but Better Performance

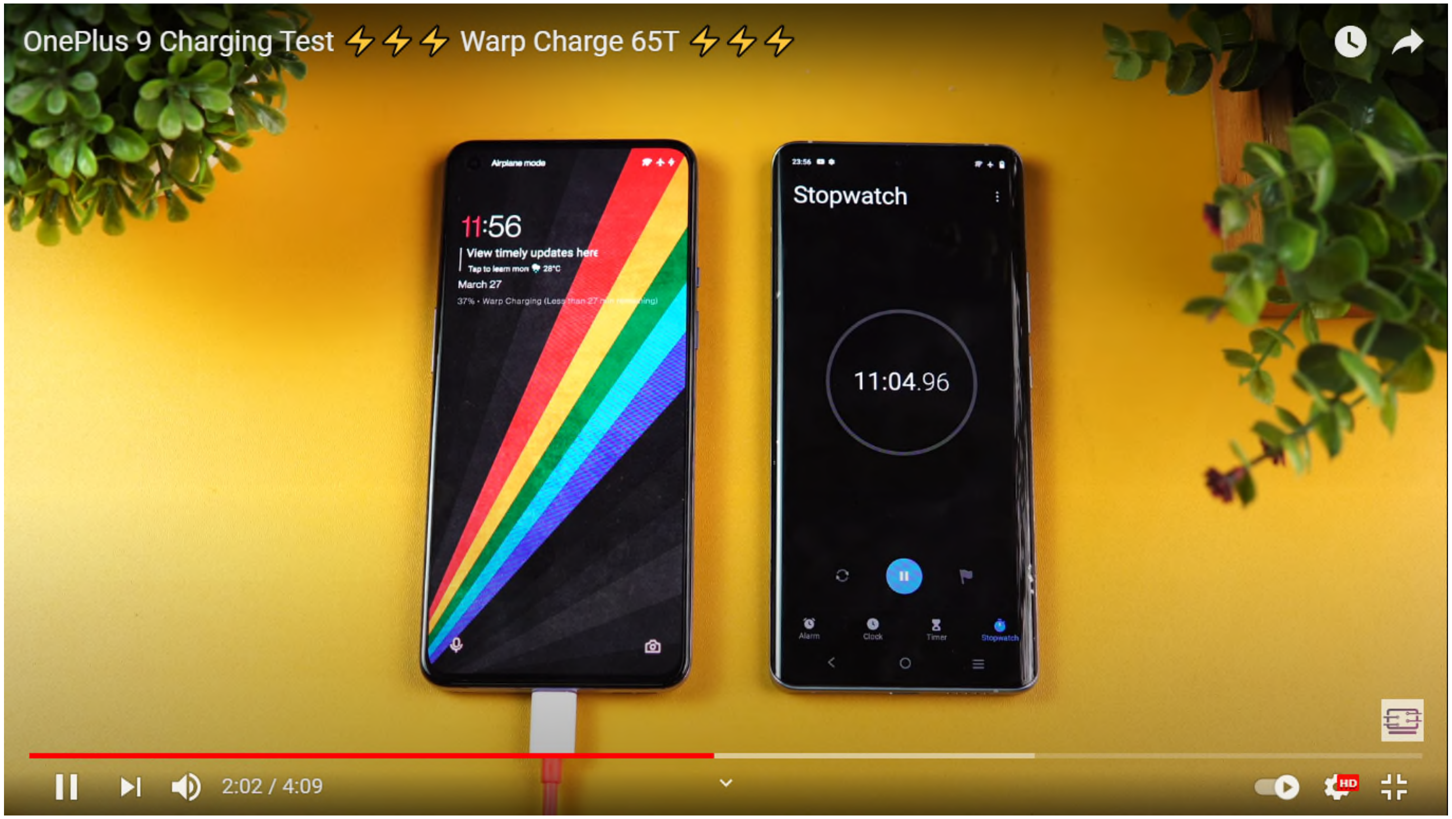
We've also upgraded wired charging on the OnePlus 9 and 9 Pro. We still use the same dual-battery layout present on the OnePlus 8T, with the two internal batteries charging at up to 32.5W, for a combined peak charging of 65W.

What's changed with the upgraded battery is that we've selected a number of upgraded components, in combination with system optimizations. Compared with the OnePlus 8T's Warp Charge 65, the OnePlus 9 and 9 Pro use a brand-new custom battery cell to reduce internal resistance, thereby also reducing battery cell heating during charges. This gives the OnePlus 9 and 9 Pro the ability to charge at a higher output for an extended period so the battery can fill up faster.

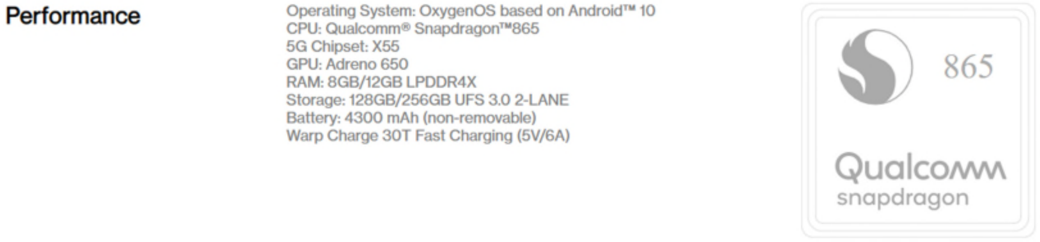
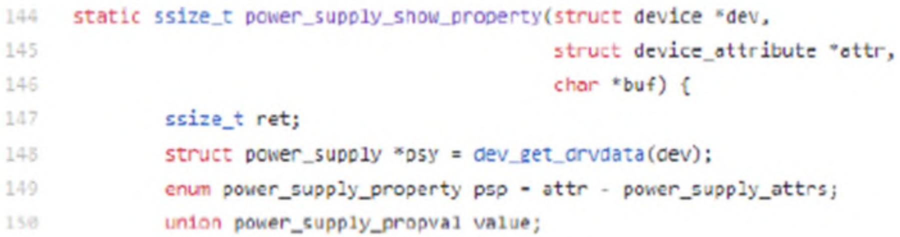
While the OnePlus 8T takes 39 minutes to charge from 1% to 100% with Warp Charge 65, the OnePlus 9 and 9 Pro only take **29 minutes to charge from 1% to 100% with Warp Charge 65T,** making it one of the fastest wired charging solutions on the market.

Fig. 43

Source: https://forums.oneplus.com/threads/deep-dive-lets-talk-charging.1403556/, Page 3, Last Accessed June 24, 2021, Exhibit L

	Citation 44: OnePlus 9 lock screen display remaining charging time estimate  OnePlus 9 Charging Test ⚡⚡⚡ Warp Charge 65T ⚡⚡⚡ Airplane mode 11:56 View timely updates here Tap to learn more 28°C March 27 37% • Warp Charging (Less than 27 min remaining) Stopwatch 11:04.96 Alarm Clock Timer Stopwatch 2:02 / 4:09 HD Fig. 44 Source: https://www.youtube.com/watch?v=CCHaMivzIvE, Time- 2:02, Last accessed June 24, 2021, Exhibit K
7. The method of claim 1, further comprising:	The accused product practices a method comprising calculating, by the apparatus, a time remaining to charge in the constant current phase based on the constant current phase charging characteristics and a time remaining to

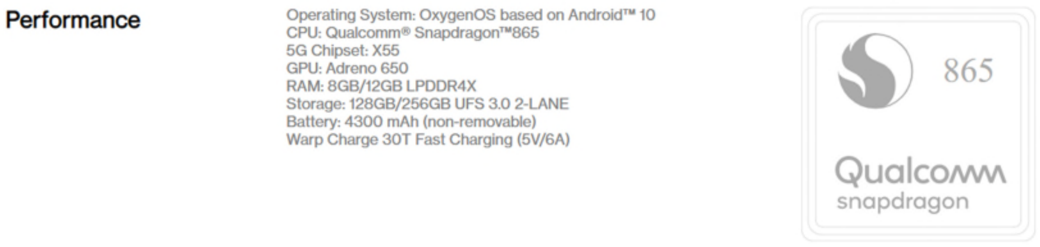
calculating, by the apparatus, a time remaining to charge in the constant current phase based on the constant current phase charging characteristics and a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant current phase.	charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant current phase. *Refer to supporting evidence of claim element 1[c] and 1[e].*
14Pre. A non-transitory computer readable medium, comprising program instructions, which when executed	The accused product comprises a non-transitory computer readable medium comprising program instructions, which when executed by a computer processor. OnePlus 8 series comprises of 5G supported Qualcomm Snapdragon 865 processor along with RAM/ROM for various storage purposes. See Fig. 45.

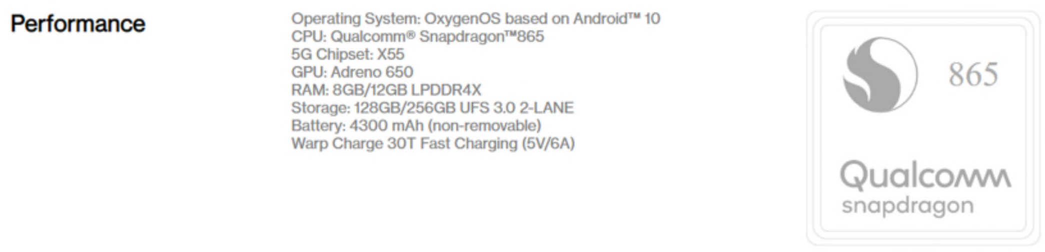
by a computer processor, perform:	**Citation 45: One Plus 8 Specifications**
	
	Performance
	Operating System: OxygenOS based on Android™ 10
	CPU: Qualcomm® Snapdragon™865
	5G Chipset: X55
	GPU: Adreno 650
	RAM: 8GB/12GB LPDDR4X
	Storage: 128GB/256GB UFS 3.0 2-LANE
	Battery: 4300 mAh (non-removable)
	Warp Charge 30T Fast Charging (5V/6A)
	
	Fig. 45
	Source: https://www.oneplus.in/8/specs, Page 1-2, Last Accessed May 18, 2021, Exhibit A
	Moreover, the accused product contains program instructions for the fulfillment of various purposes. See Fig. 46.
	Citation 46: Exemplary Program Instructions used in the accused product
	
	`static ssize_t power_supply_show_property(struct device *dev,`
	`struct device_attribute *attr,`
	`char *buf) {`
	`ssize_t ret;`
	`struct power_supply *psy = dev_get_drvdata(dev);`
	`enum power_supply_property psp = attr - power_supply_attrs;`
	`union power_supply_propval value;`
	Fig. 46

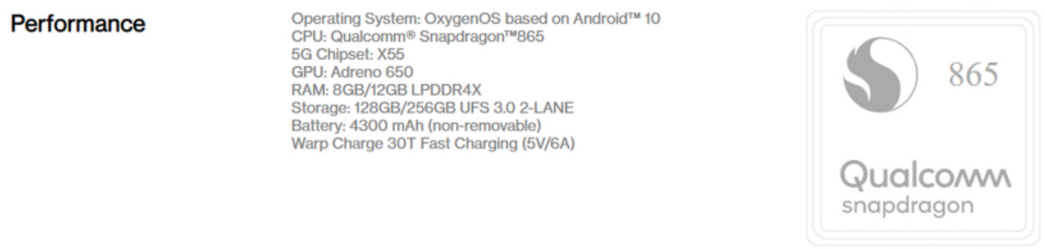
	Source: https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 2, Last Accessed May 18, 2021, Exhibit E
14a. detecting an availability of a charger adapter;	The accused product comprises a non-transitory computer readable medium comprising program instructions for detecting an availability of a charger adapter. *Refer to supporting evidence of claim element 1[a].*
14b. determining whether a battery charging point is in a constant current phase or in a constant voltage phase, based on pre-determined battery charging characteristics, wherein the pre-determined battery charging characteristics comprise constant current phase charging characteristics	The accused product comprises a non-transitory computer readable medium comprising program instructions for determining whether a battery charging point is in a constant current phase or in a constant voltage phase, based on pre-determined battery charging characteristics, wherein the pre-determined battery charging characteristics comprise constant current phase charging characteristics and constant voltage phase charging characteristics. *Refer to supporting evidence of claim element 1[b].*

and constant voltage phase charging characteristics;	
14c. calculating a time remaining to charge in the constant current phase based on the constant current phase charging characteristics, if the battery charging point is in the constant current phase; and	The accused product comprises a non-transitory computer readable medium comprising program instructions for calculating a time remaining to charge in the constant current phase based on the constant current phase charging characteristics, if the battery charging point is in the constant current phase. *Refer to supporting evidence of claim element 1[c].*
14d. calculating a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point	The accused product comprises a non-transitory computer readable medium comprising program instructions for calculating a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant voltage phase. *Refer to supporting evidence of claim element 1[d].*

is in the constant voltage phase;	
14e. wherein the time remaining to charge in the constant current phase is based on stored charge characteristics in the constant current phase, when the battery charging point is in the constant current phase and wherein battery stored charge characteristics comprises a battery stored charge value based on monitored tracking of battery charging and discharging.	The accused product comprises a non-transitory computer readable medium comprising program instructions wherein the time remaining to charge in the constant current phase is based on stored charge characteristics in the constant current phase, when the battery charging point is in the constant current phase and wherein battery stored charge characteristics comprises a battery stored charge value based on monitored tracking of battery charging and discharging. *Refer to supporting evidence of claim element 1[e].*
15Pre. An apparatus, comprising:	The accused product comprises at least one processor.

15a. at least one processor;	OnePlus 8 series comprises of 5G supported Qualcomm Snapdragon 865 processor. See Fig. 47. **Citation 47: One Plus 8 Specifications**  Performance Operating System: OxygenOS based on Android™ 10 CPU: Qualcomm® Snapdragon™865 5G Chipset: X55 GPU: Adreno 650 RAM: 8GB/12GB LPDDR4X Storage: 128GB/256GB UFS 3.0 2-LANE Battery: 4300 mAh (non-removable) Warp Charge 30T Fast Charging (5V/6A) Fig. 47 Source: https://www.oneplus.in/8/specs, Page 1-2, Last Accessed May 18, 2021, Exhibit A
15b. at least one memory including computer program code;	The accused product comprises at least one memory including computer program code. OnePlus 8 comprises RAM and ROM for various storage purposes. See Fig. 48. **Citation 48: One Plus 8 Specifications**

	 **Performance** Operating System: OxygenOS based on Android™ 10 CPU: Qualcomm® Snapdragon™865 5G Chipset: X55 GPU: Adreno 650 RAM: 8GB/12GB LPDDR4X Storage: 128GB/256GB UFS 3.0 2-LANE Battery: 4300 mAh (non-removable) Warp Charge 30T Fast Charging (5V/6A) Fig. 48 Source: https://www.oneplus.in/8/specs, Page 1-2, Last Accessed May 18, 2021, Exhibit A
15c. the at least one memory and the computer program code configured to, with the at least one processor, cause the processor at least to:	The accused product comprises at least one memory and the computer program code configured with the at least one processor. OnePlus 8 series comprises of 5G supported Qualcomm Snapdragon 865 processor along with RAM/ROM for various storage purposes. See Fig. 49**.** **Citation 49: One Plus 8 Specifications**



Performance

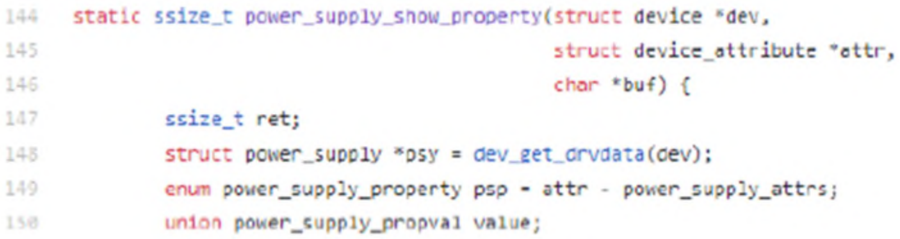
Operating System: OxygenOS based on Android™ 10
CPU: Qualcomm® Snapdragon™865
5G Chipset: X55
GPU: Adreno 650
RAM: 8GB/12GB LPDDR4X
Storage: 128GB/256GB UFS 3.0 2-LANE
Battery: 4300 mAh (non-removable)
Warp Charge 30T Fast Charging (5V/6A)

Fig. 49

Source: https://www.oneplus.in/8/specs, Page 1-2, Last Accessed May 18, 2021, Exhibit A

Moreover, the accused product contains computer program code for the fulfillment of various purposes. See Fig. 50.

Citation 50: Exemplary Computer Program Code used in the accused product



```
static ssize_t power_supply_show_property(struct device *dev,
                                          struct device_attribute *attr,
                                          char *buf) {
        ssize_t ret;
        struct power_supply *psy = dev_get_drvdata(dev);
        enum power_supply_property psp = attr - power_supply_attrs;
        union power_supply_propval value;
```

Fig. 50

	Source: https://github.com/OnePlusOSS/android_kernel_oneplus_sm8250/blob/cad09e061ef6cf689a4a1e54d27562e0f042236a/drivers/power/supply/power_supply_sysfs.c#L353, Page 2, Last Accessed May 18, 2021, Exhibit E
15d. detect an availability of a charger adapter;	The accused product is an apparatus comprising at least one processor to detect an availability of a charger adapter. *Refer to supporting evidence of claim element 1[a].*
15e. determine whether a battery charging point is in a constant current phase or in a constant voltage phase, based on pre-determined battery charging characteristics, wherein the pre-determined battery charging characteristics comprise constant current phase charging characteristics and constant voltage	The accused product is an apparatus comprising at least one processor to determine whether a battery charging point is in a constant current phase or in a constant voltage phase, based on pre-determined battery charging characteristics, wherein the pre-determined battery charging characteristics comprise constant current phase charging characteristics and constant voltage phase charging characteristics. *Refer to supporting evidence of claim element 1[b].*

phase charging characteristics;	
15f. calculate a time remaining to charge in the constant current phase based on the constant current phase charging characteristics, if the battery charging point is in the constant current phase; and	The accused product is an apparatus comprising at least one processor to calculate a time remaining to charge in the constant current phase based on the constant current phase charging characteristics, if the battery charging point is in the constant current phase. *Refer to supporting evidence of claim element 1[c].*
15g. calculate a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant voltage phase;	The accused product is an apparatus comprising at least one processor to calculate a time remaining to charge in the constant voltage phase based on the constant voltage phase charging characteristics, if the battery charging point is in the constant voltage phase. *Refer to supporting evidence of claim element 1[d].*

15h. wherein the time remaining to charge in the constant current phase is based on stored charge characteristics in the constant current phase, when the battery charging point is in the constant current phase and wherein battery stored charge characteristics comprises a battery stored charge value based on monitored tracking of battery charging and discharging.	The accused product is an apparatus comprising at least one processor wherein the time remaining to charge in the constant current phase is based on stored charge characteristics in the constant current phase, when the battery charging point is in the constant current phase and wherein battery stored charge characteristics comprises a battery stored charge value based on monitored tracking of battery charging and discharging. *Refer to supporting evidence of claim element 1[e].*
16. The apparatus of claim 15, wherein the battery stored charge characteristics	The accused product is an apparatus comprising at least one processor wherein the battery stored charge characteristics comprise data that provides a remaining charging time estimate based on a battery stored charge estimate.

comprise data that provides a remaining charging time estimate based on a battery stored charge estimate.	*Refer to supporting evidence of claim element 2.*